Vol. 4, No. 5 September/October 1986 $4.00

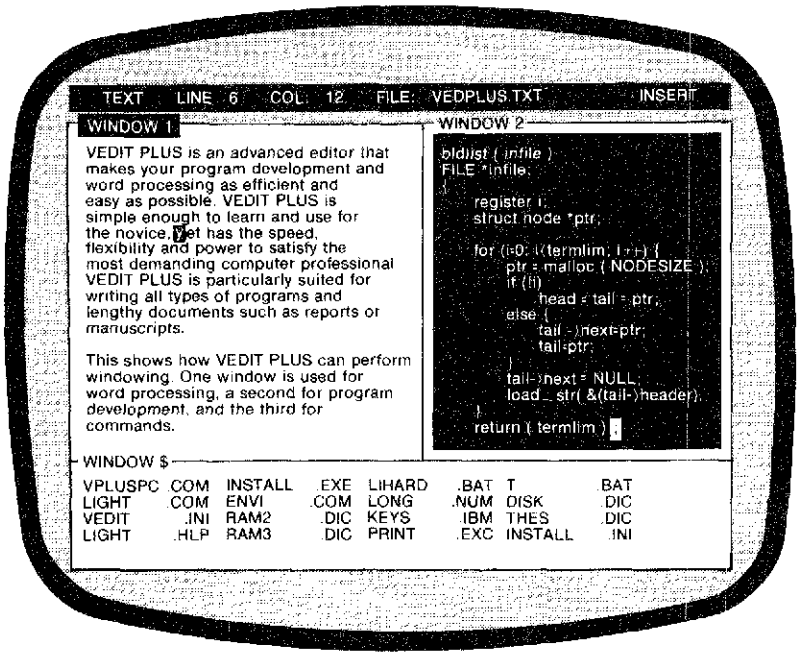
programmer's journal

The Resource Journal for IBM PC Programmers

GRAMMER'S JOURNAL 4.5

TO PROLOG, Mark Minasi
URBO PROLOG, Lew Baxter
TURBO PROLOG, Tom Swan
NEST EGGS, Paul Barkley
UTILITY, Dan Daetwyler
ATEMENT, Graham Pearson
D, PART 3, Rex Jaeschke
RTUP CODE, William Hunt
SHORT BOARDS REVIEW

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED


TEXT LINE 6 COL 12 FILE: VEDPLUS.TXT INSERT
WINDOW 1
VEDIT PLUS is an advanced editor that makes your program development and word processing as efficient and easy as possible. VEDIT PLUS is simple enough to learn and use for the novice, yet has the speed, flexibility and power to satisfy the most demanding computer professional VEDIT PLUS is particularly suited for writing all types of programs and lengthy documents such as reports or manuscripts.
This shows how VEDIT PLUS can perform windowing. One window is used for word processing, a second for program development, and the third for commands.
WINDOW 2
bldlist (infile)
FILE *infile;
{
register i;
struct node *ptr;
for (i=0; i<(termlim; i++) {
ptr = malloc (NODESIZE);
if (i)
head = tail = ptr;
else {
tail->next=ptr;
tail=ptr;
}
tail->next = NULL;
load_str(&(tail->header);
}
return (termlim) ;
WINDOW $
VPLUSPC .COM INSTALL .EXE LIHARD .BAT T BAT
LIGHT .COM ENVI .COM LONG .NUM DISK .DIC
VEDIT .INI RAM2 .DIC KEYS .IBM THES .DIC
LIGHT .HLP RAM3 .DIC PRINT .EXC INSTALL .INI

VEDIT PLUS

MULTIPLE WINDOWS

POP-UP MENUS

KEYSTROKE MACROS

EXECUTE DOS PROGRAMS

For over six years VEDIT has been the choice of professionals who demand the most powerful editing software available. CompuView has once again enhanced this power with the latest VEDIT PLUS - you can now open windows to simultaneously edit several files, access editing functions with pop-up menus, use keystroke macros to speed editing and run other programs within VEDIT PLUS.

Whether your needs are program development, technical writing or word processing, VEDIT PLUS is your answer. VEDIT PLUS is simple enough to learn for the novice, yet has the speed, flexibility and power to satisfy the most demanding computer professional. Its powerful macro programming language helps you eliminate repetitive editing tasks.

If you take your editing seriously, you need VEDIT PLUS. With over 40,000 users, you can depend on VEDIT PLUS to perform consistently and reliably. As have GE, EDS, U.S. Navy, GM, Sperry and many others. VEDIT PLUS supports color windows on the IBM CGA & EGA and even windows on most CRT terminals. Available for MS-DOS, PCDOS, CP/M-86 and CP/M-80. List price $185.

"To sum things up, VEDIT PLUS is a small, fast, sophisticated editor with a wealth of features and a good macro language. It offers many rewards for the dedicated programmer."
Computer Language, Chris Wolf, Scott Lewis, Mark Gayman 6/86

"VEDIT PLUS is a wholly remarkable program: blindingly fast, extremely powerful, and highly flexible."
Profiles Magazine, Robert Lavenda 4/86

VEDIT PLUS FEATURES

- Simultaneously edit up to 37 files of unlimited size.
- Split the screen into variable sized windows.
- 'Virtual' disk buffering simplifies editing of large files.
- Memory management supports up to 640K.
- Execute DOS commands or other programs.
- MS-DOS pathname and CP/M user number support.
- Horizontal scrolling - edit long lines.
- Flexible 'cut and paste' with 36 text registers.
- Customization - determine your own keyboard layout, create your own editing functions, support any screen size, any CRT.
- Optimized for IBM PC/XT/AT. Also 132 column & up to 70 lines.

EASY TO USE

- Interactive on-line help is user changeable and expandable.
- On-line integer calculator (also algebraic expressions).
- Single key search and global or selective replace.
- Pop-up menus for easy access to many editing functions.
- Keystroke macros speed editing, 'hot keys' for menu functions.

FOR PROGRAMMERS

- Automatic Indent/Undent for 'C', PL/I or PASCAL.
- Match/check nested parentheses, i.e. '{' and '}' for 'C'.
- Automatic conversion to upper case for assembly language labels, opcodes, operands with comments unchanged.
- Optional 8080 to 8086 source code translator.

FOR WRITERS

- Word Wrap and paragraph formatting at adjustable margins.
- Right margin justification.
- Support foreign, graphic and special characters.
- Convert WordStar and mainframe files.
- Print any portion of file; separate printer margins.

MACRO PROGRAMMING LANGUAGE

- 'If-then-else', looping, testing, branching, user prompts keyboard input, 17 bit algebraic expressions, variables.
- CRT emulation within windows, Forms entry.
- Simplifies complex text processing, formatting, conversions and translations.
- Complete TECO capability.
- Free macros: • Full screen file compare/merge • Sort mailing lists • Print Formatter • Main menu

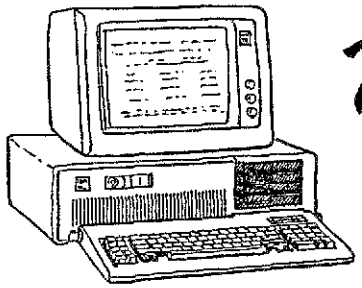
VEDIT and CompuView are registered trademarks of CompuView Products, Inc. MS-DOS is a registered trademark of Microsoft. CP/M is a registered trademark of Digital Research. WordStar is a registered trademark of MicroPro.

Circle Number 56 on the Reader Service Card

CompuView

1955 Pauline Blvd., Ann Arbor, MI 48103 (313) 996-1299, TELEX 701821

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED



programmer's journal

The Resource Journal for IBM PC Programmers

Volume 4, Number 5 September/October 1986

CONTENTS

NOTE FROM THE EDITOR 2

LETTERS 3

PJ FORUM: DTACK GROUNDED, The Junk Mail Flyer *by Hal W. Hardenbergh* 5
High-performance 32-bit PCs, the 68040, Weitek's data-path part, and fulminations on a host of other subjects

REDIRECTIONS: Frank D. Greco 10
Large screen monitors; language translators; UNIX; and more

PROLOG: An Introduction to Prolog *by Mark Minasi* 18
The Clocksin and Mellish standard

Turbo Prolog:

It's Not Real Prolog! *by Lew Baxter* 24
A programmer at Logicware rates Borland's new language

Another Miracle? *by Tom Swan* 27
A Turbo Pascal adherent looks at Borland's Prolog

CONSULTING: Tax Traps, Consulting Styles, and Nest Eggs *by Paul Barkley* 37
More practical tips for consultants

ASM: Search and Display *by Dan Daetuyler* 40
A programmer's search utility

ASM: An Assembler Programming Suggestion *by Graham Pearson* 44
Make your code modular by using the GROUP statement

C: The Proposed ANSI C Language Standard, Part 3 *by Rex Jaeschke* 46
#include and naming DOS subdirectories

C: Monitoring Stack Usage and a Guide to Startup Source Code *by William Hunt* 48
C functions that measure stack usage and a description of Lattice C's startup code

REVIEW: Short Boards 52
PJ *looks at two short board accelerators*

MISC 61

PJ Billboard 63

HOUSEKEEPING 64
Errata, Advertiser Index, Peer Review

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

programmer's journal
The Resource Journal for IBM PC Programmers

Publisher: Elizabeth Oakley
Editor: Robert Keller
Managing Editor: Erin O'Connor
Production: Ralph Bentley
Promotion: Robert Tappan
Circulation: Gary Hoyer
Cover: Ralph Bentley
Illustrations: Julia King
Production Assistance: Holly Tracy

Contributing Writers:

Paul Barkley
Lew Baxter
Dan Daetwyler
Frank D. Greco
Hal W. Hardenbergh
William James Hunt
Rex Jaeschke
Mark Minasi
Graham Pearson
Tom Swan

Founder: Greg Estes

Business and Editorial Offices: Programmer's Journal
29 West 29th Ave., Suite 5
Eugene, OR 97405
or
P.O. Box 30160
Eugene, OR 97403
(503) 484-2162

Subscription Inquiries: Programmer's Journal
P.O. Box 3000
Dept. EE
Denville, NJ 07834
(503) 484-2162

Subscriptions: Bi-monthly rates $24 U.S. ($30 in Canada and $40 elsewhere) 2 years $44 U.S. ($55 in Canada and $75 elsewhere) 3 years $64 U.S. ($80 in Canada and $110 elsewhere)

Articles and letters in Programmer's Journal represent the views of their authors and not necessarily those of the publisher. Letters to the editor become the property of Programmer's Journal and may be edited and published.

Advertising: For ad rates, deadlines, and placement, contact Ralph Bentley or Robin Tappan, (503) 484-2162, or write to Programmer's Journal, P.O. Box 30160, Eugene, OR 97403.

IBM and Personal Computer are trademarks of International Business Machines; dBASE II and dBASE III are trademarks of Ashton-Tate.

Programmer's Journal (ISSN 0747-5861) is published bi-monthly by Oakley Publishing Company. Application to mail at second-class rates is pending at Eugene, OR.

POSTMASTER: Send address changes to Programmer's Journal, P.O. Box 3000, Dept EE., Denville, NJ 07834.

© 1986

NOTE FROM THE EDITOR

Accelerators

Richard Cox writes to report that the Orchid Turbo 286e coprocessor reviewed two issues ago won't work with disk drives in an expansion chassis. Robert Hurt at Orchid confirms, saying that the problem is the wait-state inserted every time the expansion chassis is accessed. (He suggests swapping the drives into the motherboard chassis.)

Richard mentioned that Classical Technologies of the Bay Area is working on a new version of their 286 board that will cope with expansion unit compatibility.

Classical Technologies' board didn't reach us in time for this issue's Short Board 286 Accelerator review.

Orchid's short board Tinyturbo arrived accompanied by a new 286e Coprocessor board with 2 Megs on board and some interesting software. Robert Hurt tells us that you can now run a program on the host and the coprocessor at the same time. A Tinyturbo could be running one program, the 286e the other.

A priority before the boards leave town will be to try to set up a display for each. Compile on one while you work on the manual on the other—that's time sharing that makes a lot of sense.

Peer Review

Summing up the Peer Review response for issue 4.3 (the modest orange issue with the black car speeding from the page) is a challenge. People were so creative—ranking 4 articles Number 1, for example, and ranking an article 99th.

Every article but one received first place votes. The accelerator board review was first, with Paul Barkley's consulting column, Bela Lubkin's Pascal article, and George Defenbaugh's and C.D. Hawkinson's Assembler articles close behind. Congratulations.

This issue contains another Peer Review card.

PJ's SIG

PJ now has a SIG on MNEMATICS, a nationwide Videotex service that is a local phone call even here in Eugene, Oregon. For now, the registration fee (normally $40 or so) is waived if you mention PJ, and on-line time is about as cheap as you can get — $4.50 an hour or so, less than long distance rates. England and other European nations are also coming on line.

By the time you read this, V1.0 and V1.1 of the Dhrystone benchmarks reported in PJ Forum last issue should be on the SIG. Included is C source code.

386

Phoenix announces that it has shipped its AT-compatible 386 ROM BIOS to "15 major OEM customers." They say the BIOS supports processing speeds up to 20MHz, supports the Origami 387, and "takes advantage of the 386 architecture to enhance the performance of the BIOS-related function of switching between real and protected modes."

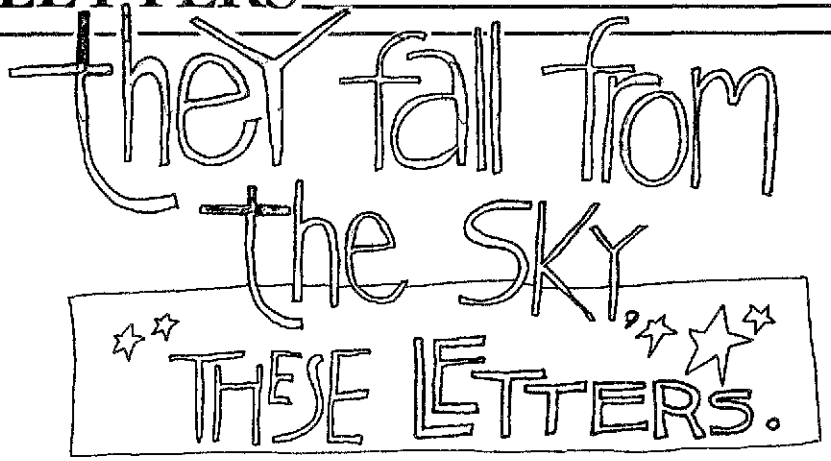
Phoenix says that the number of PCs relying on Phoenix for compatibility is approaching 1.5 million—although that includes some products besides BIOSes.

IBM has been rumbling about leaving the "commodity" computer market.

—Robert ❑

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

LETTERS



Rex to Hal

Rex Jaeschke replies to Hal W. Hardenbergh's letter (PJ, Volume 4, Number 3, May/June 1986):

The fact that most UNIX is written in C and that large numbers of users seem to be satisfied with UNIX performance leads me to believe that C is a good alternative to Assembler for certain applications. If C can be used to write a reasonably high performance operating system, I see no reason why it could not successfully implement a language compiler or interpreter. In fact, most C compilers and interpreters are written in C. I certainly don't see why C couldn't be used to write a decent and fast BASIC translator.

You complain about the size of code generated by C. Well what can I say? I certainly won't apologize for sloppy implementations. Of course you can get a very fat program as a result of coding in C, but that's not the fault of C. Most likely the algorithm is deficient, coding practices sloppy, and possibly, the compiler itself does a poor job of code generation and optimization.

The problem here is that you have experienced one or more bad scenarios with programs that happen to have been written in C. While I don't dispute your reports, I won't infer that because of those cases, the C language itself is somehow responsible. Ritchie did a fine job of designing C so that a good compiler running on most modern architectures had a realistic shot at generating very compact code. Whether or not C compiler writers have taken full advantage of his brilliance is the real question here.

Rex Jaeschke
Reston, VA 22091

P.S. By the way, I'm not a C bigot, nor do I use UNIX much. I also use a considerable amount of Assembler in my real-time, process-control consulting work. I also use an extremely fast full-screen text editor which is primarily written in DeSmet C, a well known, compact code generator.

IBM Defended; UNIX Relegated to Its Proper Place

Despite all the "kicking" IBM gets, I find few users really gripe about significant problems. The restriction on the number of files that may be open (through handles) per task is significant, yet there has been very little complaint about it. I do some "consultant" work, and I've run into a customer or two who objected to the standard keyboard. I've "replaced" the IBM keyboard with four other vendors' efforts, and in each case have been forced to "absorb" the cost because the customer preferred the IBM board after trying the substitutes.

Bickering has been going on as long as computers have existed (I can personally vouch for the last 25+ years). The facts are relatively simple. Programmers are commercial artists, rather than "hard" scientists, who react to subjective stimuli in weird and wonderful ways. The language that "agrees" with Programmer One does so because it "fits" the way his mind works and therefore seems "natural" to him. Programmer Two's mind works in a different way, so the language feels "un-natural" and awkward to him. Programmers are almost as autocratic and opinionated as surgeons, so they will quickly rationalize defenses of their "pet" languages. Few bother to narrow in on the critical elements of language selection. Although a good professional would laugh if you suggested the use of FORTRAN for a text processing application, I've heard its use strongly defended.

Incidentally, although there is much "to do" about "C," Pascal, etc., I see little on COBOL, despite the fact that it is probably the most common language in use in industry today. I don't like the language, but that doesn't change the fact that it exists and is the primary production language for industry. I suspect this is just another example of mags publishing a bunch of articles on the way "we wish it were," instead of "the way it is."

I must address one last issue, UNIX!!! This is a great development environment. It is not a particularly good "production" environment. If your readers are primarily programmers who are programming for the sake of programming, then push on for the UNIX grail. If, instead, they are programmers who are trying to develop systems for the marketplace, knock off 80% of the UNIX garbage, and let's stick to DOS. The UNIX'ers of today are the same as the set of APL'ers of yesteryear. I can rarely get a customer to read (let alone understand) the DOS manual (or even the limited set of DOS commands that he needs), and I shudder to think of turning one loose with UNIX.

Most customers buy a computer system to do a specific job within their existing operation. They do *not* buy with the intent of "getting into" programming and computer systems and resist even learning to do "keyboarding." There are exceptions, of course, but generally this attitude holds true.

Dan Daetwyler
Springdale, AR 72764 □

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Vitamin C™
A Healthy Supplement For C Programmers

Perfect Windows, Powerful Data Entry, FULLY Integrated, For Effortless Data Entry Windows!

- Complete input formatting
- Unlimited Validation
- Full attribute control
- Field sensitive application help system
- *Multiple virtual windows*
- Fully automatic, collision proof overlay and restore
- Print to & scroll background windows
- Animated window "zoom"
- Move, grow, shrink, hide, or show any window
- "Loop function" allows processing while awaiting input

and much much more!

Designed to increase your productivity and help you produce superior applications in dramatically less time! Includes 100% source, tutorial, reference manual, examples, and sample programs.

Specify Microsoft, Lattice, Computer Innovations, Aztec, or Mark Williams. Ask about Unix.

$149.95

NOW... VCScreen !

Our new interactive screen "painter" actually lets you draw your data entry windows! Define fields, text, boxes & borders. Move them around. Change attributes. Then the touch of a button generates C source code calls to the Vitamin C routines! Now, your screen designs will be faster and more pleasing when they are created with VCScreen!

Requires Vitamin C library above. For IBM & compatibles. **$99.95**

For Orders Or Information,

(214)245-6090

Creative Programming Consultants
Box 112097 Carrollton, TX 75011-2097
Include $3 ground, $6 air, $15 overnight shipping, $25 if outside USA. Texans add 6% tax. All funds must be in U.S. dollars drawn on a U.S. bank.

Circle Number 40 on the Reader Service Card

HELP

NO ROYALTIES

is at hand

HELP/Control™ - an online help subsystem for the IBM-PC. Increase product marketability. Reduce product development time.

HELP/Control is fast. Screens - up to full display size - appear almost instantaneously because the runtime system is written in assembler and accesses the screen memory directly. Smaller screens appear as windows. Once they've helped, the original display is restored just as quickly.

HELP for the programmer. A few simple subroutine calls add a full-featured online help subsystem to your application because **HELP/Control** *is a runtime system linked or loaded with that application.* **HELP/Control** has been fully tested with Microsoft C, Lattice C, Turbo Pascal, IBM BASIC (Interpreter and Compiler), Microsoft FORTRAN, IBM COBOL and assembler. It is distributed with sample programs in each language.

HELP for the documentation writer. Build screens. Define captions. Associate each caption with a screen. **HELP/Control** includes a screen generation language, HELPGEN, that reads your sources, creates a runtime file, and lets you use your favorite editor. Our **HELP/Control** reads that file and displays help screens when the user activates the help function.

HELP for the end user. Use the cursor control keys to select a caption. Press return. That's it. Each screen has a number of highlighted captions which indicate other screens with information on that subject. Lotus 1-2-3 users will feel right at home. We've even made the entire **HELP/Control** manual available as a set of screens so you can browse without ever cracking a book.

HELP/Control comes complete with a detailed manual, both online and printed, with information for the programmer and documentation writer. It also includes instructions for the end user which may be incorporated into the application documentation.

PC-DOS 2.0 or greater required for developing **HELP/Control** applications. Applications using **HELP/Control** will run under PC-DOS 1.0 or greater. The runtime system requires approximately 9K for code and buffers for full size help screens.

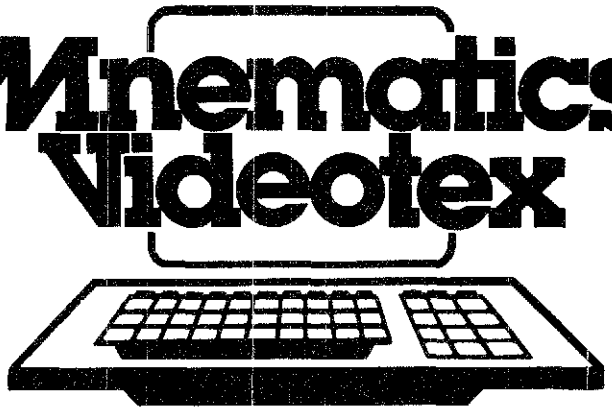
The complete *HELP/Control* package (software, both manuals, demo programs) is $125.00. A demonstration diskette, including the online manual, is available for $15.00. To order or for more information, contact MDS, Inc. at (207) 772-5436. MasterCard and VISA accepted.

MDS, INC., P.O. BOX 1237, PORTLAND, MAINE 04104

MDS

Circle Number 69 on the Reader Service Card

Mnematics Videotex



Get your **Free** password and save over 50% on usage!
Regularly 4.50 hr. non prime time 300/1200/2400 baud.
Access numbers in over 800 cities and 120 countries.
Over 50 SIG's, Databases, Games, Realtime Chat and more!

YES! Count me in!

Send me a free subscription to Mnematics Videotex.
I've enclosed 5.00 for 2.5 hrs. of connect time.

Name _____
Address _____
_____ Zip _____
Phone _____
Visa/Mastercard (circle one) _____ Expires _____

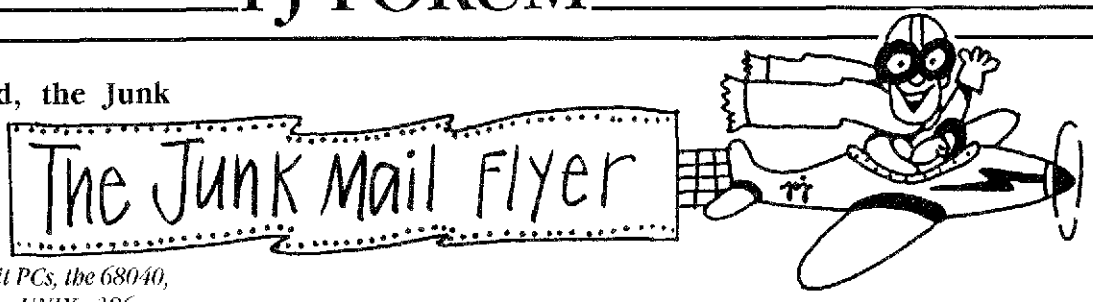
Send to Mnematics Videotex, Sparkill, New York 10976-0019

Hurry, offer expires Nov. 1st, 1986.

Circle Number 65 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

DTACK Grounded, the Junk Mail Flyer



by Hal W. Hardenbergh

High-performance 32-bit PCs, the 68040, two-tier DRAM pricing, UNIX, 386 vs. 68020, Weitek's data-path part

High Performance:

21 Apr *Electronic Engineering Times* reported the efforts of Apple, Atari, and Commodore to come up with high-performance 32-bit personal computers. Apple's is Jonathan (*Rollerball*, anyone?), Commodore's is RANGER, and Atari's is the rather uninspired TT (Thirty-Two, as in bits). *EET* reports that Atari has built *two* TTs, one with the 68020 and the other with the AT&T 32-bitter as a negotiating wedge to get lower Motorola pricing (where have I heard that one before?). In fact, *EET* also reports that Atari even built a 32032-based prototype, but immediately junked it after running a few benchmarks (that one seems familiar, too).

Everyone seems to be concentrating on a $3,000 price tag for an advanced-performance 32-bit personal computer. It would seem that for $3,000, folks expect to get a 12.5MHz 68020-based system with one wait-state and an empty socket for a 68881. Add 2 Megabytes of DRAM, an 800K floppy disk, keyboard, a few expansion slots, and a 1024 x 800 non-interlaced *color* monitor, and you have what folks seem to expect. And they expect it for $3,000? *Good luck!*

The above paragraph describes what can be produced a year from now for $5,000 by Atari or for $6,000 by Commodore or for $9,999.95 by Apple. The problem, of course, is the 1024 x 800 non-interlaced color monitor. Them things is expensive.

Those prices can be lowered only by compromising on the monitor. The lower the price, the greater the compromise. Unfortunately, it is almost certain that mass-marketing constraints will force all three companies to sell ga-ga-ga interlaced monitors. This will absolutely wreck the usefulness of the machines without saving what I would call a significant amount of money. (You do understand that $5,000 and up computers would probably be used more than 30 minutes a day—and the quality of the monitor is *vitally* important if one is going to spend much time staring at it. That's why I love my two Z-80-based Eagle IIs; their text display is *super*, much better than the genuine IBM PC display.)

128-bit 68040?

The following is from 28 Apr *EET*: "Motorola, Inc. is reportedly pegging design specifications for a 128-bit extension of the 68020 32-bit microprocessor . . . tentatively called the 68040. Motorola's designers are looking at a 128-bit-wide (four 32-bit words) internal chip structure. . . Sources say the 68040 will process four 32-bit words in one 25-MHz clock cycle. That would enable the 68040 to deliver what is effectively a 100MHz, 32-bit-processing speed/word combination."

If you expect Motorola to deliver samples of the above this fall, you must also believe in the tooth fairy. By the way, can you spot the technical error the *EET* editor made in the above? Still, the above does give you an idea of the kinds of research going on at Motorola.

The problem is that modern microprocessors are outrunning the available mass local memory (DRAMs). Even Digital Acoustics has a bunch of 68000s which are rated faster than we can ship them with affordable DRAM. And where the 68000 uses four clock periods per memory cycle, the 68020 uses three. You can now purchase 20MHz 68020s routinely in quantities of 100. Three clock periods at 20MHz is 150nsec while 120nsec DRAM is rated at a R/W cycle time of 235nsec. So a 20MHz 68020 will be slowed *at least* 57% by using 120nsec DRAM for mass storage.

This is why almost all 68020 systems use cache memory. Cache memory and its associated TLB (translation look-aside buffers) and controlling logic is *damned* expensive. So expensive, in fact, that you automatically have minicomputer-type pricing. So you might as well go ahead and add memory management so that you can time-share 32 users. . .

How fast can you go with 120nsec DRAM? Well, three clocks at 12.5MHz is 240nsec. With a R/W cycle time of 235nsec, it is theoretically possible to run a 12.5MHz 68020 with 120nsec DRAM at no wait-states. After all, you have an entire 5nsec to play with! I once read that microprocessors spend 90% of the time reading, 7% of the time writing, and 3% on I/O. The 68000's I/O is just more reading and writing, so the 68000 series reads about 12 times more often than it writes. Obviously, it is more important to speed up reading than it is to speed up writing. The members of the 68000 family that are currently in production read and write at the same (bus-bandwidth) rate. Hmmm.

Suppose Motorola put an additional 32 input-only data lines on the 68020, er, 68040? Then it could read *two* 32-bit words in one 3-clock memory cycle. To run at 25MHz, the 68000 needs to read a 32-bit word every 120nsec. . . or two 32-bit words every 240nsec. With a 64-bit input port, the 25MHz 68040 could run with 120nsec DRAM with (mostly) no wait-states *without using expensive cache memory* (and the cache's expensive ancillary circuitry).

(Old timers will recognize this as "memory interleaving," a technique that has been in common use since the 1950s in the mainframe world. What is new is that the massive amount of logic that the technique requires has disappeared into the hypothetical 68040's chip.)

For you software types, let me make clear what I am talking about: The hypothetical 68040 I just described has the same 32-bit internal CPU as the existing 68020. When the 68040 writes to memory,

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

it writes up to 32 bits, just like the existing 68020. But when it reads memory, it reads 64 bits at a time into its prefetch queue. That doubles the external bus-bandwidth for a given memory access time, and since a microprocessor reads about 12 times more often than it writes, the performance is greatly increased where we have a fast CPU and relatively slow, cheap DRAM. The trick is that the input port to the prefetch queue is 64 bits wide but the output of that queue to the CPU is only 32 bits wide.

The reason that *all* wait-states would not be eliminated is that program flow occasionally branches to a new location and we have to insert wait-states to get the first half of the 64-bit memory fetch into the CPU. *But this would not happen on common, short loops!* Remember, the 68020 has an internal 256-byte cache, so execution of short loops doesn't even use the external memory after the code has been loaded once. Result: Probably 99% of all memory fetches would not require a wait-state even when running with a 25MHz CPU! Remember, we are still going to have wait-states on the infrequent occasions when we *write* to memory.

As long as I am fantasizing about this hypothetical 68040, Motorola, would you please add a separate internal 256-byte data cache? Like the separate program and data caches the Fairchild Clipper uses? Pretty please?

A 68040 with a 64-bit input port makes a *lot* of sense in terms of bang-per-buck. I would not be at all surprised if Motorola sampled something like that late this year or early next year. *EET*'s report of four internal 32-bit processors? Not in this decade, chum. And probably not early in the next decade, either.

Name the Board:
We are not yet laying black tape on clear acetate, but we will need to give serious consideration to naming our next-generation simple 68020 board. Our present offering is the Beaucoup Grande, which is a combination of French and Spanish. I propose that the next-generation board name be comprised of three words, which should be German, Urdu, and Basque, respectively. Any nominations?

68881 Report:
The local Motorola microprocessor gurus very kindly took James and me to lunch. One had just returned from Austin, with the earliest report that there had just been a new, successful mask for the 68881, which apparently eliminates many of the bugs in the old mask. Incidentally, they tell me there was only one serious bug in the old 68881. Although the bug list was long, almost all of the bugs involved errors in the least-bit(s) position. In other words, the results were largely accurate, which is why some companies were indeed shipping products with 68881s.

The Return of Two Tiers:
Back in DTACK #42 I wrote about the fall of the two-tier DRAM pricing. The second tier is now almost completely reconstructed. 256K DRAMs are now about $2 in Japan and about $4 in this country (the exact price depends on quantity, speed, and source). *Whoopee, we have protected the profits of a few American chip-makers!* Is this a cause for celebration?

Cheap Gasoline! Hooray!
Hmm... With oil under $15 a barrel, exploration for oil has flat stopped.

Digital Acoustics has this customer which had developed a better mouse trap. When exploring for oil, a lot of seismological data is processed on site to evaluate the possibility and location of producing wells. In the past, one placed a VAX 11/780 in an 18-wheeler and trucked it to the site. VAXen need to be air-conditioned, so one places a large air conditioner in a second 18-wheeler and a diesel generator to power the compressor in a third 18-wheeler. *Mucho dinero!* Finally, the computer terminals arrive in a van.

We have been telling you for some time that a DTACK GRANDE with no wait-states and one or two math chips in a 3-holer board is just about equivalent to a VAX 11/780. That happens to be the truth; after all, the 11/780 is a vintage 1978 design.

So our customer developed software to run on our board set that formerly ran on a VAX 11/780. And our board set fit inside the van, dispensing with three 18-wheel trucks and their expensive contents. There was a lot of excitement last fall when they announced this product. We had envisioned their being a significant customer in 1986. But then Saudi Arabia turned on the oil pumps.

So we'll have to look elsewhere this year for a source of income to replace that which has been lost. Would you like to buy HBASIC for your 1040ST, maybe? Geez, even a dinky little outfit like us being affected by world events...

UNIX Pricing:
Certain readers of my former newsletter have vociferously assured me in past years that UNIX was really going to make it big in the low-end personal computer marketplace *just as soon as the price comes down*! Well, those readers will undoubtedly be overjoyed at two advertisements in the April *UNIX REVIEW*. One, on page 39, lists a 68000-based MEGADATA UNIX system for only $3,299. 1,000 systems are available at that price. A smaller ad on page 66 offers 30 of what is apparently the same make for only $3,250. Since both ads pointedly mention how many systems are available, it appears that these are closeout sales, as in "thank you for your money and good-bye forever." A question for my readers who know more than I about low-end UNIX: Just exactly how low does the hardware price have to get before we can decide whether *you* were correct or I was correct? *Seventeen cents?*

Speaking of 17:
Convergent Technologies is now delivering the last of those 50,000 UNIX PCs to AT&T, which has neatly placed 40,000 of them on seventeen miles of shelves. Where they are gathering dust. I can't imagine why. After all, for about $8,000 you get a unit with 512K RAM, which is a very useful automatic telephone dialer. Ah, well, AT&T is a big outfit and can easily afford to write off 40,000 units after its independent auditors notice those units ain't selling.

Let's see now: 40,000 units times $4,000, which is about what AT&T paid for those turkeys, is $160 million. *Ka-flush!* Right down the old tubes. Fortune Systems, which may be misnamed, has already run through more than $60 million. That's about a *quarter billion dollars* lost by just *two* of the companies pursuing low-end UNIX! So every time you pick up *BYTE* magazine, you see yet *another* company planning to make it big in low-end UNIX! Mama mia! *When will people learn?*

UNIX Done Right:
There are companies, Sun Microsystems and Charles River, for instance, which turn a nice profit selling $50,000 UNIX systems. I understand that these are very good systems, but at $50,000 I am not going to buy one at FEDCO and take it home with me. (Single-user systems are priced as low as $27,000.) Now, that is obviously the marketplace where UNIX belongs. 3,000-page manuals in the mass personal computer marketplace? Ho ho ho!

UNIX systems are traditionally used by

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

programmers and documenters, for whom computing is what happens while the computer waits for the next keystroke. Sun Microsystems is going after the engineering marketplace, where the concept of computing is what happens *after* the last keystroke. How are they doing that? Well, each user gets a 68020 with multi-megabyte memory for his very own. These *very* intelligent (!) terminals are then networked into a file-server with a big hard disk. Now, that is *not* UNIX, which by definition (AT&T's) is a system in which many users share one CPU. But Sun calls it UNIX because so many people know for a fact that UNIX is wonderful...

Gee, 68020 with multi-megabyte memory for one's very own. Doesn't that sound nice? I wonder if I can get hold of one of those things for about $25,000 less money and with 2,800 fewer manual pages? Which runs an incrementally compiled BASIC that is written entirely in assembly? Where I can get the source code to that BASIC? Is this what junk mail writers call subtle advertising?

Records Done Wrong:
PC Tech Journal's review of six BASICs (June 1986) compared the time to load and save 30,000 bytes of data. HBASIC running on an Apple or IBM host with our new high density floppy drives can do that in about 3 seconds. If MS DOS were properly written, it could in principle do the job in 6 seconds; the way the disk drivers are actually written with interleaving, it would take 12 seconds, except that MS DOS is written incorrectly, and hence, one uses PRINT and INPUT statements to store and load numeric data to and from disk. And so the average benchmark time for the six alternate BASICs was 24.7 *minutes*! BASICA did the job in 9 *minutes*. I repeat HBASIC can do that in 3 *seconds* on a floppy disk!

Two-Tier *Oops!*
A funny thing happened just as the new two-tier DRAM pricing scheme was being put into place. It turns out that, at the urging of T.I., the legal proceedings against the Japanese DRAM makers were specifically limited to imports from Japan. All of a sudden, the Japanese companies do not ship direct to the U.S. What they do now is ship to Taiwan or Indonesia for the final stages of packaging and testing, and then ship directly to the U.S. And, believe it or not, that means that six months of court wrangling is for nought, *nada!* It is as if the U.S. had done nothing at all! And the U.S. cannot suddenly impose penalties on shipments from those other countries because of a teensy technicality called "due process." Tsk.

80386 = Intel>64K:
You have all been reading about the onslaught of 386-based machines that will arrive late this year. *Wrong!* What will arrive late this year is an onslaught of 386 *product announcements*. Six months later some of those will turn into real products, all of which will die immediately when IBM brings out its bus-incompatible machine. (Sam, *please* do not inundate me with clippings of those 386-based product announcements. Instead, *check their availability*!)

32-Bit Merchant Micros:
There are some 32-bit micros which are proprietary; HP has them, DEC has them, IBM has them, NCR has them, DG (the other one) has them. We are here talking about the stuff not being sold as a proprietary product.

I think that the 68020 series (and its successors) are going to divide the market with Intel's 386. Except for the IBM connection, the 68020 might dominate. It was here *first*, and by all accounts it performs extremely well. The INMOS Transputer appear occasionally in specialized applications, doing digital signal and image processing and the like. As far as chips are concerned, that is about it. Everything else is much too little or too late.

There are a couple of board-level CPUs which will fight it out in the non-proprietary super-micro market, just above the 68020. One is the Fairchild Clipper, and the other is the new MIPS series of products. MIPS is offering products at several performance levels. The higher performance levels require the use of MIPS memory boards. (These are *not* products for the bang-per-buck personal computer market.)

Tightly Coupled Memory:
Both the Fairchild and the MIPS products outperform the 68020. Both use a method of improving the CPU-mass memory interface. In Fairchild's case, both program and data caches are used *and* a method of using the "page mode" of conventionally priced DRAM to improve the bus-bandwidth to the mass memory. MIPS, in its high-end products, has obviously gone beyond that. I suspect that MIPS has done some memory interleaving with the support logic migrated to custom VLSI. They have probably also included some "page mode" tricks.

C & PASCAL PROGRAMMERS

Blaise Computing provides a broad range of programming tools for Pascal and C programmers, with libraries designed for serious software development. You get carefully crafted code that can be easily modified to grow with your changing needs. Our packages are shipped complete with comprehensive manuals, sample programs and source code.

C TOOLS PLUS
$175.00
NEW! Full spectrum of general-purpose utility functions; windows that can be stacked, removed, and accept user input; interrupt service routines for resident applications; screen handling including EGA 43-line text mode support and direct screen access; string functions; and DOS file handling.

PASCAL TOOLS/TOOLS 2
$175.00
Expanded string and screen handling; graphics routines; easy creation of program interfaces; memory management; general program control; and DOS file support.

VIEW MANAGER
$275.00
Complete screen management; paint data entry screens; screens can be managed by your application program; block mode data entry or field-by-field control. Specify C or IBM/MS-Pascal.

ASYNCH MANAGER
$175.00
Full featured asynchronous communications library providing interrupt driven support for the COM ports; I/O buffers up to 64K; XON/XOFF protocol; baud rates up to 9600; modem control and XMODEM file transfer. Specify C or IBM/MS-Pascal.

Turbo POWER TOOLS PLUS
$99.95
NEW! Expanded string support; extended screen and window management including EGA support; pop-up menus; memory management; execute any program from within Turbo Pascal; interrupt service routine support allowing you to write memory resident programs; schedulable intervention code.

Turbo ASYNCH PLUS
$99.95
Complete asynchronous communications library providing interrupt driven support for the COM ports; I/O buffers up to 64K; XON/XOFF protocol; and baud rates up to 9600.

RUNOFF
$49.95
NEW! Text formatter written especially for programmers; flexible printer control; user-defined variables; index generation; and general macro facility. Crafted in Turbo Pascal.

EXEC
$95.00
Program chaining executive. Chain one program from another even if the programs are in different languages. Shared data areas can be specified.

ORDER TOLL-FREE 800-227-8087!

BLAISE COMPUTING INC.
2560 Ninth Street, Suite 316 Berkeley, CA 94710 (415) 540-5441

Circle Number 2 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Atron's PC/AT Bugbusters

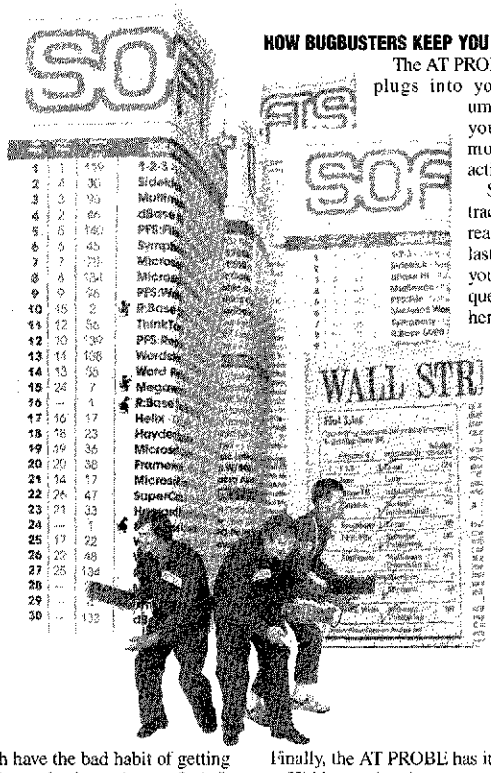
Hardware-assisted Software Debuggers for Bullet-proof PC/AT-based Products

A BUGBUSTER STORY

Brad Crain, a project manager at Software Publishing (the people who developed both PFS:WRITE and PFS: FILE), relates the following: "On Friday, March 22, 1985, I was about to get on an airplane with Jeff Tucker, who was co-author of PFS:WRITE with me, and fly to IBM's Boca Raton, Florida facility. For a week, we had been unsuccessfully trying to isolate a bug in a new software product. In a last, desperation move, I set up an early-Saturday morning appointment with ATRON.

"Three of us walked through ATRON's door at 8:00 the next morning. Using ATRON's hardware-assisted debugging tools, we had the problem *identified* and fixed by 10:30AM."

Mr. Crain concludes: "We'd never have found the bug with mere software debuggers, which have the bad habit of getting over-written by the very bugs they're trying to find. It doesn't surprise me that almost all the top-selling software packages were written by ATRON customers. Now that they've broadened their PC family of debuggers to include a PC/AT debugging tool, those of us seriously into 80286 development are greatly relieved."



HOW BUGBUSTERS KEEP YOU FROM GETTING SLIMED

The AT PROBE is a circuit board that plugs into your PC/AT. It has an umbilical which plugs into your 80287 socket and monitors all processor activity.

Since AT PROBE can trace program execution in real time, and display the last 2048 memory cycles, you can easily answer the questions: "How did I get here?" and "What *are* the interrupts doing?"

It can solve spooky debugging problems. Like finding where your program overwrites memory or I/O - impossible with software debuggers.

You can even do source-level debugging in your favorite language, like C, Pascal or assembler. And after your application is debugged, the AT PROBE's performance-measurement software can isolate your application's bottlenecks.

Finally, the AT PROBE has its own 1-MByte of memory. Hidden and write-protected. How else could you develop that really large program, where the symbol table would otherwise demand most of your PC/AT memory.

ARE YOU TRYING TO DO SOMETHING SCAREY?

Like developing your AT-based software product in the dark? Without professional debugging tools?

Seven of the ten top-selling software packages listed by the *THE WALL STREET JOURNAL** were produced by ATRON customers. The PC PROBE™ bugbuster ($1595) accounts for much of this success. Now that the PC/AT is the new standard for advanced commercial and scientific development, ATRON is proud to announce the AT PROBE™ bugbuster ($2495). It has even more debugging capabilities than the PC Probe.

BORLAND'S PHILIPPE KAHN: "THERE WOULDN'T BE A SIDEKICK™ WITHOUT ATRON'S DEBUGGERS."

So why waste more time reading though your program listing for the ten thousandth time, trying to find why your program starts howling with every full moon. Be like BORLAND, get your Atron bugbuster today and bust bugs tomorrow.

atron
THE DEBUGGER COMPANY
20665 Fourth Street • Saratoga, CA 95070 408/741-5900

*WSJ, June 24, 1985, reporting Softsel figures. © 1985 by ATRON. PC PROBE™ and AT PROBE™ ATRON SIDEKICK™ Borland IBM Corp. owns numerous trademarks. All by TRBA

Circle Number 1 on the Reader Service Card

RESIDENT
MACRO ASSEMBLER

In the next few years, the bus-bandwidth to the main memory is going to become *the* issue which determines performance!

I have just finished reading about one of Weitek's new products, and the damn thing is plain scary. They call it a data-path part. It contains three orthogonal 32-bit ports, a 32-bit floating point multiplier, a 32-bit signed floating point adder (which they call an ALU), some logic for performing divide without external support, and a lot of stuff that used to go on a board to support separate multipliers and adders. (The three 32-bit ports are *not* just your conventional two-operand in plus result-out ports; this chip is *much* more flexible than that!) This chip—the WTL3332—contains an internal register file of 32 ea 32-bit registers. It can perform, on a pipelined basis, one triadic instruction (e.g., A + B + or-C = D) in a single clock cycle, or 100nsec. Now, assuming that A, B, or C is already in one of those 32-bit registers, *just how are you going to move two 32-bit operands in and one result out every 100nsec using commodity DRAM without some fancy stuff such as interleaving?*

You are looking at 20 32-bit megaflops in a single chip with lots of internal support logic for just $425/100ea! (Stuff like this is one of the reasons I am not optimistic about the future of the Transputer. This is also why some folks are calling the 68881 slow!) *EET*'s Stan Baker reports (2 Jun) that Weitek designed that chip in less than a year, compared to ". . . the near-farcical delays in getting MMUs and coprocessors for Motorola's 68020. . ." (Stan picks on the other major houses too; Motorola's no Lone Ranger.)

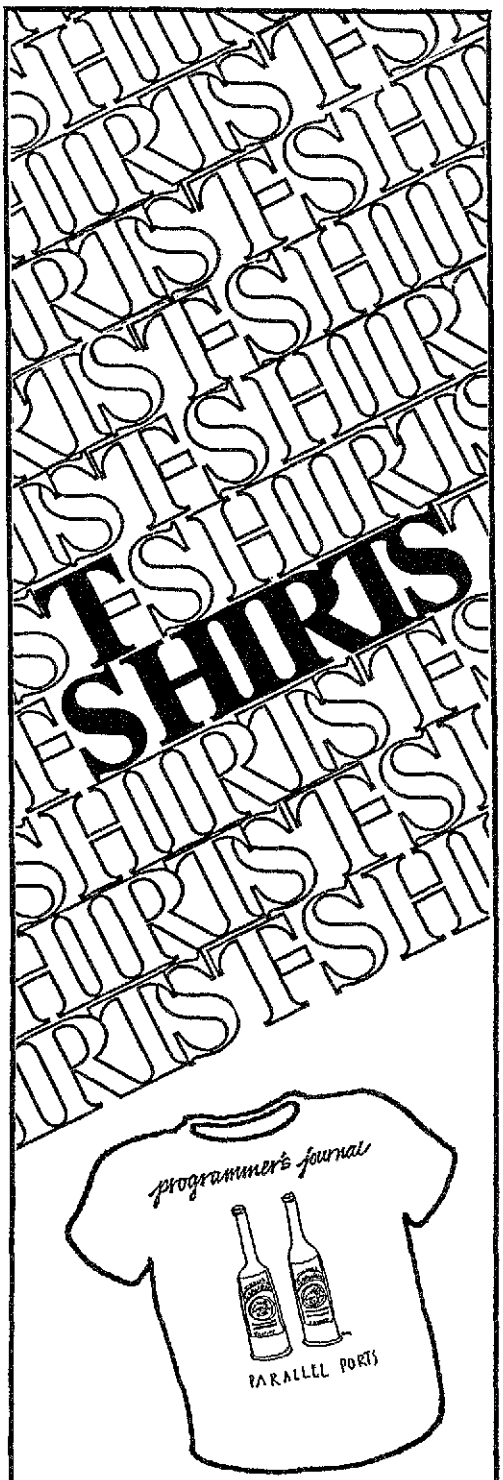
Short Stuff:

Now I know why I was unable to obtain the old Victor organizational chart, which featured Chuck Peddle at the top, 500 circles at the bottom, and 500 lines going direct from Chuck to each of those 500 circles. The reason I was unable to obtain it was that another Silicon Gulch personal computer company grabbed it and is using it, although not all of those 500 circles at the bottom are filled yet.

If you think you might want a 2 Megabyte piggyback expansion board for your Beaucoup Grande, let us know so we'll know how many to build on the first run. □

DTACK Grounded
1415 E. McFadden, Ste. F
Santa Ana, CA 92705

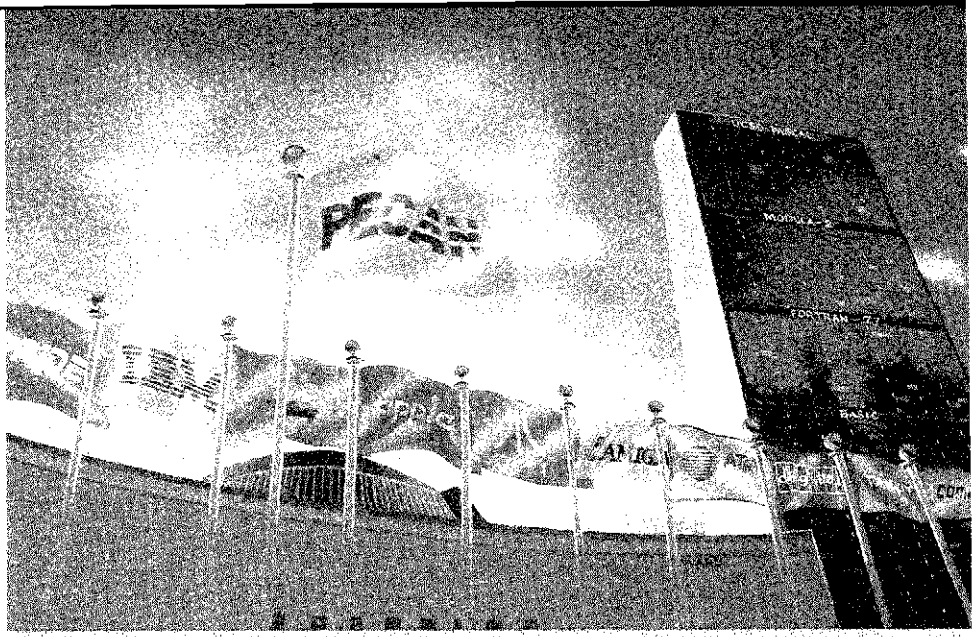
PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED



PROGRAMMER'S JOURNAL

Famous Cartoons Reformatted as T-Shirts

See inside back cover for details.



Super Power Compatibility

The POWER SYSTEM™ from PECAN has set a new standard for comprehensiveness in program development environments that you won't find anywhere else:

PORTABILITY: Whether you're an ardent beginner or a seasoned pro, THE POWER SYSTEM lets you design software that can run on virtually any mini or micro without having to rewrite the program. The program you write on one computer will operate on another— even if the two hardware systems are incompatible. Your software never becomes obsolete — even if your hardware does.

INTEGRATABILITY: The versatility and flexibility of THE POWER SYSTEM enables different users to exchange program modules freely with one another, even though the modules may be written in different programming languages.

LANGUAGE INTERCHANGEABILITY: THE POWER SYSTEM lets you customize the modules of your application in the most appropriate of Pecan's powerful languages including UCSD PASCAL, BASIC, FORTRAN-77, ASSEMBLER, or MODULA-2. Once you've created your multi-language program, the modules are linked automatically. The application then runs as if you'd written it all in one language.

THE POWER SYSTEM FEATURES:

• NATIVE CODE • SEPARATE COMPILATION OF UNITS • EXTENDED PRECISION ARITHMETIC • DYNAMIC MEMORY MANAGEMENT • EVENT-DRIVEN MULTI-TASKING • CONFORMANT ARRAYS • PROCEDURAL PARAMETERS • 8087 AND BCD SUPPORT (PC's AND COMPATIBLES) • LARGE FILE EDITOR • LIBRARY UTILITY • ON-LINE HELP SYSTEM • FILE MANAGER

THE POWER SYSTEM Only $99.95

For PC-DOS, MS-DOS, AMIGA, ATARI ST, MACINTOSH, RAINBOW, TANDY, STRIDE, as well as most popular 8/16/32 bit systems. Also available in VAX, UNIX, PDP-11 OS's & others.

THE POWER SYSTEM includes the language of your choice: UCSD PASCAL, MODULA-2, BASIC, ASSEMBLER and FORTRAN-77. Choose an additional language for only $79.95. Buy three and we will send you a fourth one *FREE!*

To custom-tailor software, PECAN offers such programming tools as:

CROSS-ASSEMBLER/NCG PACKAGE	**$79.95**	**SOFTEACH TUTORIAL**	**$49.95**
KEYED SEQUENTIAL ACCESS METHOD	**$59.95**	**MENUMAKER**	**$49.95**
CODE OPTIMIZER	**$59.95**	**PROGRAM ANALYSIS TOOLKIT**	**$59.95**
SYMBOLIC DEBUGGER	**$59.95**	**SORT UTILITY**	**$59.95**

Write for complete list of development aids.

ATTENTION CORPORATIONS: Call and ask about our training seminars.
SCHOOLS: Special Educational Discounts

NOT COPY PROTECTED

Mail Your Check Or Money Order To:
Pecan Software Systems, Inc.
1410 - 39th Street
Brooklyn, New York 11218
(718) 851-3100
ITT TELEX NUMBER: 494 8910
COMPUSERVE CODE: 76703,520

PECAN™
The UCSD Pascal Company

Credit Card Orders Call Toll Free **1-800-63-PECAN**
(NYS) 1-800-45-PECAN

Please add $2.50 for shipping within the US. Foreign orders add $10.00 and make payment by bank draft payable in US dollars on US bank. New York State residents add appropriate sales tax.

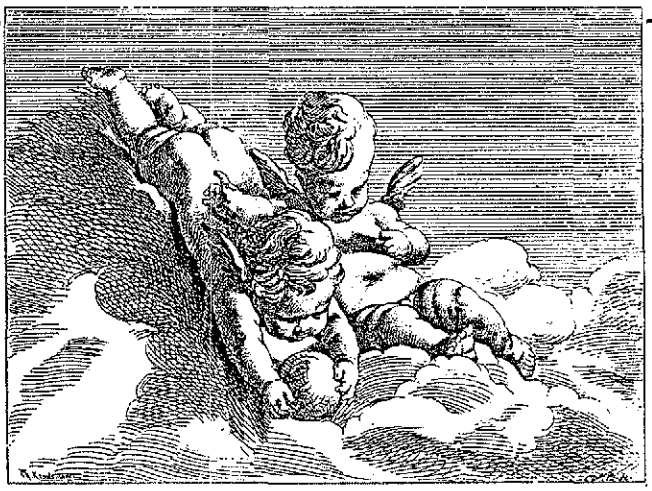
UCSD Pascal is a trademark of The Regents of University of California.

Circle Number 71 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Frank D. Greco

Large screen monitors; language translators; Jonathan, Amy and Ranger; UNIX; Apple, National Geo, and Lucasfilms; reader responses, and more



Small Windows Are a Pane in the Glass

For many PC users, a window manager such as Microsoft's Windows or Digital Research's GEM (let's all forget about Topview!) does not make sense on an 80 character by 24 line display. The graphics resolution doesn't make a difference—the screen is just physically too small. Next time you see a Microsoft Windows advertisement with at least three pseudo-concurrent (ugh . . . tiled) windows open, look at how busy the screen is. There is too much information—text or iconic—on the little PC screen. The reason UNIX workstation users can't live without (ahh . . . overlapping) windows is that Sun, Cadmus, Silicon Graphics, Masscomp, and Apollo (and others) all have 19-inch, bit-mapped, high resolution (actually considered medium resolution in the UNIX workstation industry) screens. Window managers just don't cut it on the normal small PC monitor.

So just as graphics systems are spurring mouse sales (do you think it's mere coincidence that Microsoft manufactures a mouse?), window managers (and window-able programs) are going to spur sales of large screen monitors (and, of course, more mice). Once you use a physically larger monitor (17 or 19 inches—measured diagonally, of course) with high resolution graphics capability, you'll never want to squint at your PC screen again. I occasionally use a text-only Wyse terminal with a high resolution character set connected to a UNIX system, and thanks just to its 14-inch screen I can go for hours at a stretch without getting bleary eyed. It's certainly welcome after the small PC screen sold by IBM.

Windows on the Whirled

Enter Micro Display Systems' (800-328-9524 or 612-437-2233) Genius VHR full page monitor and controller board. This is a vertical display monitor; that is, it is higher than it is long (like a regular CRT rotated 90 degrees). It is capable of displaying 66 lines of text and 728 x 1008 *monochrome* graphics. In addition, the Genius monitor can emulate both the IBM CGA and the MDA by putting the monochrome emulation in the top half of the monitor and the CGA emulation in the bottom. Imagine the nifty debuggers you could write with this thing! The board and monitor has a slightly-pricey suggested retail of $1795. I got wind of this monitor through some talk of it on the UNIX network, Usenet. Some of the Microsoft boys on the "Net" were raving about the way the portrait-format Genius ran their Windows package. The only "problem," if you want to call it that, is that not all software packages support the slick 728 x 1008 graphics mode. The word processors that do support the Genius 66-line mode include WordStar, Volkswriter, Palantir Word Processor, Word Perfect, PC-Write, Final Word, and XY-Write. Other packages are being ported.

The software that takes advantage of the Genius hi-res graphics mode includes Microsoft's Windows series, DRI's GEM series, Media Cybernetics' Dr. Halo and HaloScan, AutoCad, and the spiffy Metawindows from Metagraphics (which I was told looks *mahvelous*).

Mine's Only Nine Inches

One other note on the 15-inch Genius display relates to some desktop publishing software. In the MDS brochure, I noticed that the slick Aldus Pagemaker package, which is selling a lot of Macintoshes lately, is scheduled for a 2nd quarter MS DOS port. The Genius monitor, along with Microsoft's Windows, will allow Aldus to give the user a *full-page* hi-res WYSIWYG view of the pre-lasered document. That's something Mac owners would envy (the Mac's screen is only nine inches). And it should light a fire under the much-acclaimed UNIX-based Interleaf to lower their prices. I would keep my eye out for this product; it should be a doozy.

Wyse Gyse

The people at Wyse (408-433-1000) have come up with a high resolution 15-inch monochrome monitor/graphics board called the Wy-700. It offers impressive graphics resolution of 1280 x 800. Clever CGA emulation is performed in four shades of gray. The character cells (text mode) at 16 x 32 pixels outdo even the IBM PC Monochrome Display Adapter (MDA). The list price is $1,595 (street price is about $1,000), which includes the monitor and the controller board. If only they'd added EGA emulation and a brighter tube. . .

Great Graphics

Microfield Graphics (503-626-9393) has announced volume production of their high resolution graphics board for the IBM PC AT. It consists of a programmable bit-sliced processor and 500 Kilobytes of on-board RAM. The T4 is capable of 1024 x 800 pixels, 4-plane (16 colors or shades) graphics. All graphics are executed quickly in hardware. The spec sheet says that polygon fill is performed at 30 nanoseconds per pixel, implying that a complete monochromatic (considers only bits on/off) screen fill would take about .024 seconds! Other key items in this product are IBM CGA emulation and a microcoded Virtual Device Interface (VDI). The microcoded VDI in the T4 should deliver enough "zip" to your programs to make it appear that your software is addressing the graphics hardware directly.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Erik the Red Would Be Erik the Dull-Gray on This Monitor

If you want even more resolution for your PC, the Viking I graphics-display subsystem offers displays up to 1280 x 960 pixels on their 19-inch monitor. The board has 256K*bytes* of monochrome display memory (their ads describe it as 2 Mega*bit* display) and uses the Hitachi HD63484 Advanced CRT Controller (ACRTC) for hardware graphics. Although it looks promising, the Viking I is not software compatible with any existing PC-bus display adapters. However, since the Viking I comes with a frame grabber, it can accept video signals from a CGA, MDA, or Hercules card, digitize them, and put the image on its large screen. In my opinion, this is kludgy; but if you need the high resolution for a custom application, the high-end high-priced ($2195) Viking I is available from Moniterm Corporation (612-935-4151).

I Can C, I Can C!

Most of us have heard of Rapitech Systems' pricey ($4,500) Fortrix-C product that converts 90% of a FORTRAN program to C code (the remaining 10% is system specific stuff—Fortrix just flags it as untranslatable). I'm not quite sure of the benefits of this product. Anyway, now they have announced a similar product called Coblix-C (guess what it does . . .). Yup, it converts COBOL to C at about 600 lines per minute. Rapitech says that it currently costs the same as Fortrix-C, but both are due to rise to $6,000. Like some other vendors' packages (BASTOC comes to mind), these packages literally translate from one language to another with minimal inference mechanisms to take advantage of the target language's capabilities. For example:

The FORTRAN segment:

```
          :
      DO 1000 I = 1, 500
          IF ( NUM(I) .EQ. TARGET ) GO TO 1500
          COUNT = COUNT + 1
1000  CONTINUE
          :
1500  CONTINUE
          :
```

converts to the following C code:

```
      for ( i = 1; i <= 500; i += 1 )
      {
          if ( num[i] == target ) goto label1;
          count = count + 1;
      }
          :
label1:
          :
```

Note that the goto's were retained and the additions of one to the variable "count" and to the loop index "i" were kept as additions and not converted to the faster increments. The translation above is merely a matter of matching syntax for syntax. It's obvious to most of you C programmers out there that perhaps a better (is anything really *optimal*?) translation should be:

```
i = 1;
while ( i <= 500  &&  num[i] != target )
{
     count++;
     i++;
}
```

You can argue that the "goto" version may generate faster code (cycle counters, start your engines), but for maintenance you'll probably want the structure that the C language is capable of.

Implying an Inference

What's needed here is an AI-based inference mechanism that can inspect the code for "patterns" and generate intelligent translations. I have worked on such a beast. Before the proverbial plug was pulled, we had a syntax-directed system that read in program code (e.g., C code) and, using grammatical rules, converted it to an intermediate tree representation. Then, using a different grammar and some inferencing algorithms, we just traversed this tree and spit out syntax for another language (e.g., Ada). Imagine taking dBASE input and putting out Informix syntax. How about taking Imagen's Impress graphics language and converting to Adobe's Postscript page description language. If you are government-minded, consider the sizzle of C to Ada translations. If you're interested in this type of stuff, Carnegie Mellon University is doing some really neat research in this area.

Jonathan, Amy, and Ranger: a JAR-ring Look

No, they are not the names of an Amish couple and their dog—these are the code names for the upcoming workstations from, respectively, Apple, Atari, and Commodore. They all use the full 32-bit 68020 chip, to Motorola's delight, and are all strongly rumored to have UNIX as an option. Many people I know would scoff at the idea that these three vendors would be any competition for the business market (i.e., IBM). But if anything, they will push what is *expected* from a small (relatively inexpensive) computer, which will indirectly push IBM to improve their boxes. At the recent Trenton State Computer Fair in New Jersey, I saw Atari ST and Commodore Amiga software that just made my jaw drop. Not that I have not seen software like that before, but the prices were *cheap*! Here I have a loaded IBM PC in front of me, and I can't do half the things I could using these smaller computers. And to boot, these *other*

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

computers sell for about one-third of what I paid for my Big Blue Box.

Jonathan is the Apple of His Eye

Apple CEO John Sculley has been making the rounds lately saying that Apple will offer a UNIX option, most likely on the Jonathan. What puzzles me is the latest cooperative agreement between Apple and Cadmus, the UNIX workstation manufacturer. Cadmus has been selling a product that emulates the Macintosh iconic interface on their UNIX-based workstations. Why would Apple sign a deal with Cadmus that at first glance competes with the Macintosh? Perhaps Apple (or Cadmus) wants to port the Cadmus product to the 17-inch, bit-mapped monochrome screen of the Jonathan. That would give Apple instant UNIX connect-ability which, besides IBM PC compatibility, Sculley desperately wants (and needs). After all, one can't get lucrative *government* deals without UNIX these days. The one hangup in this strategy is the version of UNIX that Cadmus uses on its machines. It's based on Berkeley 4.2 UNIX; the government mandates System V. Although BSD 4.2 and AT&T System V are compatible enough for most applications, the bosses in Washington say, "It gots to be System V."

Amy, Whatcha Gonna Do?

My sources say that Jack Tramiel is having an on-again/off-again affair with AT&T corporate about whether the upcoming Amy will be OEMed to the phone company. Apparently, Jack and his developers are eyeing those cheap AT&T 1 Megabit RAM chips but are not too enamored of the AT&T manufactured WE32X00 family of processors that AT&T wants Amy to have (along with UNIX, of course). I don't blame Jack. Have you ever tried getting WE32X00 assembly language information from the phone company? Responses I've received range from, "There is only a little manual" to, "Why do you want to program in Assembler—do it in C." There is already a sea (. . . sorry. . .) of 68K programmer developers out there waiting with open arms. These new Amy series machines are also going to have custom VLSI graphics (1024 x 1024 resolution in color) and sound chips similar to the Amiga according to an interview in *Electronic News* with Shiraz Shivji, Vice President of Research and Development for Atari.

So while a contract *has* been signed between the two As ("AT&Tari"), which microprocessor will drive the Amy has not been totally agreed upon. Unless AT&T makes Jack an offer he can't refuse, my money's on the 68020. One of my sources tells me that AT&T wants Jack to incorporate the AT&T *patented* (I still can't believe it!) "layers" window software currently found on the AT&T 5620 terminal.

The Loaned Ranger

Commodore's updated Amiga machine, code-named the Ranger, is reported to have 1024 x 720 color (hardware) graphics, 1 Megabyte of RAM upgradeable to 4 Meg, 5 internal slots, a 3 1/2-inch microfloppy, 20 Megabyte winchester disk, and an SCSI port. Even though Commodore is losing money rapidly ($130 million this year), its creditors are banking (pun intended) on the Amiga's making it big. The Amiga is probably one of the reasons Commodore received an extension on its bank loans. The Ranger is supposed to be announced in late 1986, according to strong rumors, but I hope that they slow down a bit and work out the bugs and clean up the documentation. They certainly don't want it to be like the rush job they obviously did on the original. Some blatant flubs: Some Amigas have the OS in ROM, some have it in RAM; system documentation is just now available at well known bookstores; Commodore went through delivery/new-announcement/new-delivery with their PC compatibility box. First it was a software PC emulation for about a hundred bucks, then it was a hardware emulation, then it was a coprocessor-in-a-box (the Sidecar), then the price of PC compatibility was raised to about $2,500 (for that price, I'll just go buy a second machine). Slowwwww downnnnn, Fellas. Apparently Commodore wants the Ranger to be profitable for the Christmas season. But as far as I can see, the Ranger is not a consumer item (judging from the hardware alone, it has to be priced in the $3,000 range) and does not fall into the seasonal purchasing patterns of the smaller C-64 or C-128 (which are selling like the proverbial hotcakes—2 million units in 1985). Despite their problems (and their reputation), I sincerely hope Commodore pulls it off. I'd hate to see the impressive little Amiga bite the dust. Every computer hack I know who has used the machine has fallen in love with it. Even John Draper, a.k.a. Captain Crunch, praises the thing to the skies on the "Net" (Usenet—net.amiga).

Pink Noise

Sigh of the Tiger

One casualty of the recent massive layoffs at Exxon is state-of-the-art research in UNIX bit-mapped workstations. Really inventive and interesting work in how to communicate cleanly between processes that are manifested in windows was going on. Most of the work was done on Sun Microsystems boxes and IBM RTs. This innovative "Inter-Window Communication" was described in a recent paper at the Denver Usenix conference. I guess this group just ran out of gas. . .

Autos Made by UNIX

Wow, that sounds weird. . . As of February 1986, the number one car (and computer system) manufacturer General Motors (GM) has adopted UNIX System V (or a certified equivalent) as the standard OS for manufacturing-based computer systems. So besides having UNIX-powered Suns and Apollos in their CAD departments, GM dictates the use of the UNIX operating system for automobile production systems along with MAP (Manufacturers Automation Protocol) as the communications standard. Another large design win for UNIX. For those of you out there who still think UNIX will never make it big, think again.

That Does Not Compute

I'm sure most of you have heard of Thinking Machines's new 1,000 to 7,000 MIPS parallel processor Connection Machine. It looks like a Cray-burner for specific types of applications. Besides breaking the BIPS barrier, the machine is housed in an attractive high-tech, translucent, smoke-colored enclosure. On the inside of the box, you can see a large grid of LEDs flickering on and off while the machine hums along. When I saw a picture of the machine, I wondered why the thing had lights like something out of an old sci-fi movie. I was told by a close associate of Thinking Machines head honcho Danny Hillis, that when Danny finished the design, he exclaimed, "It gotta have lights to look cool." So he rigged up a matrix of LEDs connected to a random number generator. The lights are all

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Hollywood! When Danny was asked what his goals were in creating the Connection Machine, he said, "I want to build a machine that would be proud of ME."

Baudy Details

It seems that a lot of people are bandying the phrases "baud rate" and "bit rate" around as if they were synonyms. The baud rate of a modem is not necessarily the same as the bit rate. For example, the Bell 212A is a 1200 *bit per second* modem and a 600 *baud* modem. The definition of baud rate is the rate that the communicating modems individually signal each other; that is, it is a symbol rate. In the case of the Bell 212A, a "symbol" is two bits (hence the 1200 bps vs. 600 baud). These symbols are manifested by phase shifting the frequency of the carrier signal. It's just a technicality, but then again, isn't this a technical journal?

Bittersweet Memories

Quietly, AT&T acquired a 10% stake in a small Scottsdale, Arizona, company called Thesys. Thesys makes memory chips. Why would a giant such as AT&T, who already manufactures memory chips through its AT&T Technologies division, be interested in Thesys? Well, this company already makes a $700 8 Megabyte memory card (yes, I did say *card*, not *cards*) for 80286 computers using a proprietary technology (investors *love* proprietary ideas!). Imagine what happens to the price of this *one*-slot card when it's produced in quantity! AT&T may plan to connect Counterpoint's UNIX workstations (another company in which they have equity) and Thesys memories. However, from what I heard, the president of Thesys is currently in a power battle with the company's financial backers. Apparently the venture capitalists who funded Thesys' R&D are not happy with the progress made so far.

Acorn Has a Tough Nut to Crack

Acorn Computer announced a 32-bit microprocessor that is a hopeful attempt to woo Intel and Motorola customers over to their camp. The RISC-based chip is called the VL86CO10 and is rated at 4 MIPS (whether that is a RISC or CISC MIPS rating, I don't know). You may be initially ambivalent about this processor (another me-too chip), but the neat thing about the Acorn chip is its price. Samples are available for $99. Production quantities might bring the price of this chip down to $25 within one year. Just for reference, samples of the newer NEC 32/16 bit V60 are $350. Of course this prediction presumes that Acorn's chip wins some design-ins with some major manufacturers and that some decent system software is developed for it. I'll relay more info on this chip as soon as I get more information on the VL86CO10 (whew. . . that's hard to type) from Acorn. I understand AT&T Bell Laboratories has a prototype microcomputer and an experimental minicomputer using this RISC chip in their Holmdel, New Jersey, labs. Apparently AT&T is planning a few squirrely things with Acorn.

Window Shopping

Apparently the overlapping window manager for the (yawn) UNIX PC was designed and implemented by Graphic System Software (GSS). It seems that AT&T looked all over the country for a decent window system before deciding on GSS. Unfortunately for GSS, AT&T has "sold" only about 10,000 UNIX PCs (of which 7,000 were for AT&T internal "sales"). I wonder

PC/VI

Full Screen Editor for MS-DOS (PC-DOS)

Looking for an Ultra-Powerful Full-Screen editor for your MS-DOS or PC-DOS system? Are you looking for an editor FULLY COMPATIBLE with the UNIX*VI editor. Are you looking for an editor which not only runs on IBM-PC's and compatibles, but ANY MS-DOS system? Are you looking for an editor which provides power and flexibility for both programming and text editing? If you are, then look no further because **PC/VI** IS HERE!

The following is only a hint of the power behind **PC/VI:** English-like syntax in command mode, mnemonic control sequences in visual mode; full undo capability; deletions, changes and cursor positioning on character, word, line, sentence, paragraph or global basis; editing of files larger than available memory; powerful pattern matching capability for searches and substitutions; location marking; joining multiple lines; auto-indentation; word abbreviations and MUCH, MUCH MORE!

The **PC/VI** editor is available for IBM-PC's and generic MS-DOS based systems for only $149. For more information call or write:

Custom Software Systems
P.O. Box 678
Natick, MA 01760
617-653-2555

The UNIX community has been using the VI editor for years. Now you can run an implementation of the same editor under MS-DOS. Don't miss out on the power of **PC/VI!**

*UNIX is a trademark of AT&T Bell Laboratories.

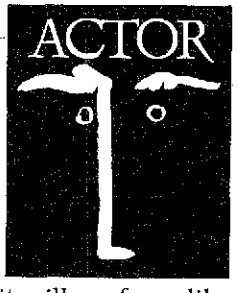
Circle Number **66** on the Reader Service Card

ACTOR debuts in one month.

There's a better way to program coming in October.

It's like an advanced research system. Instead of data structures and procedures, it uses active objects. But it produces finished, useful programs. Not experiments.

It's easy to optimize, and it has a new way to reclaim unused memory. As a result, it runs so fast on microcomputers, it's good even for real-time control.



And it's built around Microsoft® Windows, which help when you write your programs. And when other people use them.

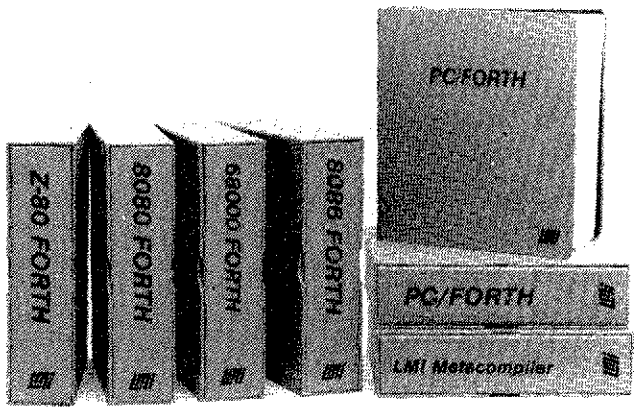
It's called ACTOR,™ and it will perform like nothing you've ever seen. To find out more about it, just call.

The Whitewater Group®
906 University Place, Evanston, IL 60201
(312) 491-2370

Circle Number **73** on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

TOTAL CONTROL
with *LMI FORTH*™



For Programming Professionals:
an expanding family of compatible, high-performance, Forth-83 Standard compilers for microcomputers

For Development:
Interactive Forth-83 Interpreter/Compilers

- 16-bit and 32-bit implementations
- Full screen editor and assembler
- Uses standard operating system files
- 400 page manual written in plain English
- Options include software floating point, arithmetic coprocessor support, symbolic debugger, native code compilers, and graphics support

For Applications: Forth-83 Metacompiler

- Unique table-driven multi-pass Forth compiler
- Compiles compact ROMable or disk-based applications
- Excellent error handling
- Produces headerless code, compiles from intermediate states, and performs conditional compilation
- Cross-compiles to 8080, Z-80, 8086, 68000, 6502, 8051, 8096, 1802, and 6303
- No license fee or royalty for compiled applications

For Speed: CForth Application Compiler

- Translates "high-level" Forth into in-line, optimized machine code
- Can generate ROMable code

Support Services for registered users:

- Technical Assistance Hotline
- Periodic newsletters and low-cost updates
- Bulletin Board System

Call or write for detailed product information and prices. Consulting and Educational Services available by special arrangement.

Laboratory Microsystems Incorporated
Post Office Box 10430, Marina del Rey, CA 90295
Phone credit card orders to: ***(213) 306-7412***

Overseas Distributors.
Germany: Forth-Systeme Angelika Flesch, Titisee-Neustadt, 7651-1665
UK: System Science Ltd., London, 01-248 0962
France: Micro-Sigma S.A.R.L., Paris, (1) 42.65.95.16
Japan: Southern Pacific Ltd., Yokohama, 045-314-9514
Australia: Wave-onic Associates, Wilson, W.A., (09) 451-2946

Circle Number 63 on the Reader Service Card.

if unbundling UNIX (literally) and the window manager developer toolkit had anything to do with it. . . .

Sun of Amiga

Look for UNIX mavens Sun Microsystems to announce a 19-inch color 68020-based (16.67 MHz clock) workstation with 1152 x 900 resolution and 256 colors with graphics hardware assist by August or September, to be priced around $15,000 (with 4 Megabytes RAM, 71 Meg hard disk, 1/4-inch tape backup, 2 serial ports, *complete* UNIX software, Network File System, SunView environment, SunWindows toolkit, CGI and CORE graphics, Ethernet capability, optional 68881 floating point chip, and, of course, a mouse). Also expect the $29,000 68020 gray-scale machine with 256 shades of gray to be lowered to about the same price at the end of the year. Using the gray scale machine, WYSIWYG-style editors will be able to integrate beautiful renditions of b&w photographs. I'm sure the newspaper people out there are just waiting for the thing. By the way, the rumor that the old Sol machine was built at Sun is not true.

Kahn Do

What was interesting about the Turbo Prolog announcement back in March was Borland's Philippe Kahn's comments concerning the Turbo Prolog user interface. He said that the interface to Turbo Prolog (series of manipulatable text windows) would be the standard for all other Borland language products. Philippe is currently porting Turbo Prolog to his personal home computer, a Sun Microsystems 68020 box.

Disk Intensive Program

As reported in the *Wall Street Journal*, an unusual *menage à trois* is embarking on a joint research program to explore new optical disk technologies for storing data and short films on microcomputers. The three companies, Apple, National Geographic Society, and Lucasfilms Ltd., hope to combine their strengths to harness the potential of CD (Compact Disk) technologies. I like the threesome: Apple has the computer technology (especially for non-computer types) and experience in the education department, National Geographic has the bottled information, and Lucasfilms has the, well, pizazz to make it sell. I hope it isn't another encyclopedia application. . . boring. . .

A Micro Microphone

Researchers at the Technical University of Darmstadt in West Germany have come up with two remarkably small microphones. The capacitor and the piezoelectric chip microphones each measure 1.7 by 2.0 by .6 millimeters. Even at this small size, there is enough room to build a signal processor on-chip! The applications possibilities are interesting: Imagine the tiny hearing aids that can be implanted directly into the ear, or the James Bondian eavesdropping devices, or the ultra small cassette recorders, or the small sonar devices that can be built with matrices of these things (robotics people take notice).

Graphical Utopia

Amongst the computer graphics wizards at the upcoming ACM SIGGRAPH annual conference in Dallas, is none other than

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Todd Rundgren of the "rock" (. . . I guess. . .) band Utopia. He will be a lecturer on the topic of integrating music and graphics. Interestingly enough, he is listed as working for Alchemedia Productions of Woodstock, New York.

This Performer Could Have a String of Hits

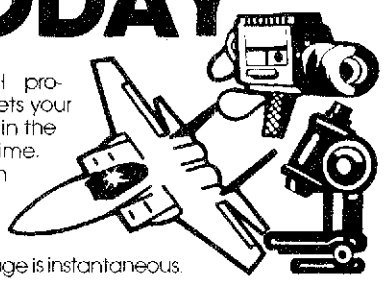
The PF474 Proximity Filter chip from Proximity Devices Corporation in Fort Lauderdale, Florida, can find the degree of similarity between strings of bytes at a rate of 49,600 byte comparisons per second and an even faster rate of 3,870 null-terminated-127-byte strings per second. The Proximity device stores a user supplied "query string" consisting of at least three parameter bytes: weight (reflects the importance of a character in a string), bias (whether the character is more important near the beginning or near the end of the string), and compensation (the dissimilarity between two characters). After doing the comparison, the PF474 maintains information about an ordered list of the "best" matches. This would be a great chip to enhance the performance of a Prolog compiler or interpreter because Prolog does a whole lot of pattern matching (besides tree traversal). It would also be a great chip to incorporate in some optical scanning applications, for example, scanning an incoming digital image for bit differences (security systems). I'm sure this chip will be of interest to those CD-ROM/CDI interface developers out there; it may just drastically reduce the tremendous amount of storage that must be allocated to the indices of the CD mega-database (in other words, just use brute force). When I called Proximity, a salesperson told me they do a tremendous business with the U.S. government. He was quite nervous when I asked about their applications; he said that they were just "proprietary."

Abort and Retry Department

I really got beat up when I said in a recent Redirections column (*PJ*, Vol. 4, No. 2, March/April 1986) that the Motorola 68020 had memory management on-chip. Robert Knight of Multi Solutions, Inc. (the S1 OS creators) and *PJ*'s own Hal Hardenbergh both point out that the 68020 does not have memory management on-chip like the iAPX 286 and 386 and Zilog's Z80K. Robert points out that "gross memory protection could be provided by the (old) 68451 MMU. You only get 32 pages, and the page size must be a power of two! With some tricky software you could fake a real MMU. Motorola has announced a better one but I don't know of anyone who actually has it." By the way, the new MMU from Motorola is called the MC68851 paged-memory management unit (PMMU). According to the Motorola specs, it makes the logical to physical translation in 45 nanoseconds; which is just enough to do it under one cycle on a 16.67 MHz 68020 system. If my math is correct, it takes 60 nanoseconds for one CPU cycle using a 16.67 MHz part. Thanks, Gents, for the correction; that's what I get for writing this column at 3:30 in the morning.

In reference to the interesting S1 operating system, Robert also points out, "S1 isn't a UNIX lookalike. The copy command is called "copy," not "cat" or "mv." And the system interface, . . . wouldn't lock anyone into a particular hardware configuration. We use the identical source for a full screen editor on both the Stride 440 with a VT-100 lookalike and an IBM AT, for example." I didn't mean to imply that S1 would lock anyone *into a particular* hardware configuration (from the sales literature and articles that I've read, Multi Solutions is stressing portability). My conjecture reasoned that if IBM had

TOOLS FOR TODAY



The MasterFORTH programming system gets your application running in the least amount of time. How? With a built-in user interface. With an interactive debugger. With a resident assembler–linkage is instantaneous.

MasterFORTH's target application generation system (TAGS) can produce a ROM-able image in seconds. Its unique symbol table identifies seldom-used code so you can pack the most power in the least amount of memory.

Choose your development system. We support the IBM PC family, Apple's Macintosh, CP/M systems and others. Need to program a controller from your IBM PC? No problem. We have cross-development sources for most popular microprocessors. You can even generate a MasterFORTH system for your target application, add a terminal and run it there. Forth is widely used in industry and aerospace for robotics and machine control.

Call or write us for details. **MICROMOTION**

8726 S. Sepulveda Bl., #A171
Los Angeles, CA 90045

(213) 821-4340

Circle Number **70** on the Reader Service Card

REAL - TIME GRAPHICS ANIMATION

For the IBM PC, XT, AT or compatible, with Color Graphics Adapter and color monitor.

With this plug-in accessory, your PC can produce graphics as fast as expensive dedicated graphics work stations. The card acts as a graphics coprocessor to the CPU. Graphics primitives are drawn by an HD63484 chip. No need to compute the adresses of individual pixels in a line, the card takes the ends of the line as input then draws the line at high speed. The line appears on the screen at a million color pixels per second.

Keep your color graphics adapter and standard RGBI monitor, this card uses both. Since your old adapter remains a part of the system, you are sure that your machine remains completely IBM compatible. Gives you all 16 colors on the screen at the same time, even with 640x200 dots.

Are you tired of having to program graphics applications in assembler and C? This card comes with an include file for Turbo Pascal. Nearly 70 procedures and functions allow you to use a high level language to write fast graphics programs. In addition to the usual lines, circles, ellipses, rectangles, 2-color area fills, etc., this package includes support for real-time animation. You can swap in a hidden screen during vertical retrace. Your animation will be clean and synchronized - no hash the easy way. Automatic checkerboard dithering is supported, giving 256 psuedo colors on a standard IBM color monitor.

Envious of your friends with Amigas and STs? This card can redraw a 21x21 3-D net graph in 1/60 sec. That is 50400 vectors per second. And the card includes support for 3-D figure drawing and rotation.

It even works with your speed up cards and fast mother-boards. No memory addresses are used, and only four bytes of I/O ports are needed, so the board won't conflict with your other graphics adapters or peripherals.

This card is available only directly from the factory. Two cards are available, one with a resolution of 640x200, the other one goes to 640x400 but requires a long persistence phosphor monitor. Both cards come with a 30 day money-back trial period. Try the card, if it doesn't perform as claimed, or if you are unsatisfied with it for any reason send it back for a full refund. The cards also come with a 180 day warranty, and include free software updates.

EHGCp2	640x200	only	$374.00	Check or Money Order
EHGCp4	640x400	interlace	578.00	30 day trial 180 day warranty (714) 261-1725
	Shipping and Handling		4.00	

E-Heart Engineering / 18103 Sky Park South #D / Irvine, CA 92714

Circle Number **67** on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

something like $1, they would probably add features to make it unique to IBM, thus "lock(ing) the user into our (IBM's) hardware." Look at the PC RT's port of UNIX, AIX, and you will see what I mean. And I'm not quite sure that I agree that the names of utility programs guarantee operating system lookalike-ability. I've written "ls," "cat," "rm," "mv," and a few other popular UNIX- based utilities for my slow-moving PC XT, but that does not make PC DOS a UNIX lookalike. Thanks for writing, Robert, and thanks again for the 68020 correction.

Don't Nix PC/IX

Another letter, from Mark E. Davidson of Chattanooga, Tennessee, was published in *PJ*, Vol. 4, No. 4, July/August 1986. Mark wrote: "I received my first issue of *PJ* yesterday, and just wanted to tell you that I really enjoyed your column "Redirections". . . . I am a PC/IX owner and would like to see [IBM] upgrade the product from System III. It is a fine port of UNIX. . . , and I have been very satisfied with it. I would hate to think that I spent quite a bit of money on a product, only to have IBM let the product die after a while." First of all, thanks a bunch for the kind words, Mark; secondly, you are not the only one satisfied with PC/IX. I chair a UNIX SIG group in New York City, and we have several happy PC/IX owners (although they weren't too happy with the $900 price tag). Most are "using the asynch ports and UNIX's multi-tasking to monitor incoming data from simultaneous external events" (. . . the method to my madness is that you can't say you're using PC/IX multi-user. . . it's licensed only for single-user).

It might be interesting, and possibly useful, to find out how and why any *PJ* readers out there are using UNIX (or a derivative/lookalike) on IBM PCs (XTs, ATs, or RTs) or compatibles. Describe your experiences, and I'll summarize the most interesting in an upcoming "Redirections" column.

Quote of the Month

Seen in a recent article in the *Wall Street Journal*: In response to the Convergent Technology and AT&T agreement that allows Convergent to sell the slow-selling UNIX PC (alias the AT&T 7300 or the 3B1), AT&T says the UNIX PC is not selling too well (actually sales are abysmal) because the UNIX PC is "ahead of its time." Yeah, right.

By the way, if you call any of the companies mentioned in "Redirections" for information, please mention that you read about their product in *Programmer's Journal*. □

Readers interested in contacting me can do so at the following address:

Frank D. Greco
c/o Programmer's Journal
P.O. Box 470
Kearny, New Jersey 07032

LUCID, INC.

Learning UNIX/C In-Depth

"FOR THE CLARITY YOU EXPECT FROM TRAINING"

Ask about our HANDS-ON course for "C" Program Development Under MS/DOS"

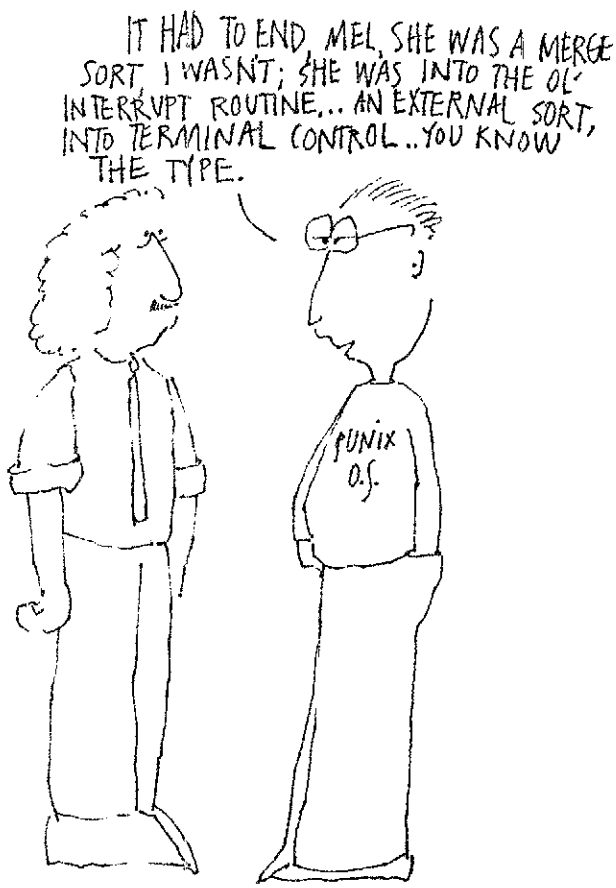
For Free Brochure, Information or Registration Please Write or Call: Alice Moss — Director

LUCID, INC.

260 Fifth Avenue
New York, N.Y. 10001
Suite 901

(212) 685-7106

Unix is a trademark of AT & T Bell Labs
MS-DOS is a trademark of Microsoft Corp.


IT HAD TO END, MEL, SHE WAS A MERGE SORT I WASN'T; SHE WAS INTO THE OL' INTERRUPT ROUTINE... AN EXTERNAL SORT, INTO TERMINAL CONTROL.. YOU KNOW THE TYPE.
UNIX O.S.

Circle Number 59 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

The 4th Personal Computer Faire presents

MEGAMAN

TRUE LIFE Adventures OF "The Man Who Became a COMPUTER CRUSADER through COMPUTER FAIRE Magic!"

I KNEW ONLY ONE THING COULD SAVE US...

AS I WALKED THE AISLES FESTOONED WITH THE LATEST COMPUTER PRODUCTS, I COULD FEEL A NEW POWER INSIDE ME UNLOCK...

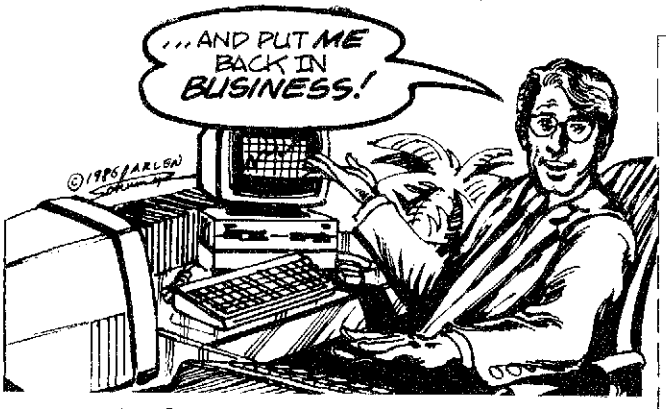
IT GREW AS I SAT IN ON IN-DEPTH SEMINARS...

AND AFTER TESTING EQUIPMENT IN THE EXHIBITS AND PRODUCT TRAINING CENTER, IT HAPPENED... I BECAME...

MEGAMAN

AND SUDDENLY I HAD MY HANDS ON ALL THE SOLUTIONS I NEEDED...

I FLEW BACK TO THE OFFICE — I PUT THOSE NEW IDEAS AND EQUIPMENT TO WORK...



Sept. 25-Sept. 27

BROOKS HALL AND CIVIC AUDITORIUM
SAN FRANCISCO! *Civic Auditorium Entrance Only*
99 Grove Street

Fill out & mail with check(s) by Sept. 5, 1986. Photocopy this form, fill out and enclose checks for additional registrants. Admission $12 in Advance; $15 at the door.

*Name & Title*_____

*Company*_____

*Address*_____

*City*_____ *State*_____ *Zip*_____ *Phone*_____

Make checks payable to "4th Personal Computer Faire." Send to: Attendee Registration, 4th Personal Computer Faire, 300 First Avenue, Needham, MA 02194.
Advance registration accepted only with full payment. Tickets mailed separately to each person.

4th Personal Computer Faire

PJ S/O

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Circle Number 58 on the Reader Service Card

An Introduction to Prolog

by Mark Minasi

Prolog is a language that many of my programming friends "would like to get around to learning, *but* . . ." (there are lots of "but"s): "It's so slow." "A full implementation costs so much." "It gobbles up RAM." "There are no good books out on it." Since Borland has announced Turbo Prolog, though, a lot of those people have decided that maybe it's time, after all, to learn Prolog.

Prolog and Artificial Intelligence

Artificial Intelligence (AI) can be implemented in any computer language. Some languages exist, however, which make some AI tasks easier. One such is Prolog (short for "PROgramming in LOGic"). Others are LISP, SNOBOL, Smalltalk, and the veritable gang of expert system development languages that are cropping up.

Some argue that LISP is a more powerful language for AI than Prolog. Despite the hours of pleasure one can derive from participating in such arguments, I shall demur.

Prolog was invented by French computer scientist Alain Colmerauer at the University of Marseilles in 1972. Like all good computer languages, Prolog is diverging into an annoying number of variations. The first was the University of Edinburgh's implementation by David Warren. A version for microcomputers, called micro-Prolog, was developed at the Imperial College of Science and Technology in London, England. Prolog's big claim to fame these days is that the Japanese have chosen it for their Fifth Generation project. We can fairly safely assume that once the Japanese are finished with it, it will look quite different from its current implementations.

There have been two main lines of Prolog: the "micro-Prolog" camp and the "Clocksin & Mellish" camp. The m-P camp use a version of Prolog that looks suspiciously like LISP, but was implemented for use on small machines. It is generally marketed with a front-end routine called SIMPLE that makes the programs much more readable. The C&M camp use the language as described in *Programming in Prolog* by Clocksin and Mellish, published by Springer-Verlag. Personally, I like the C&M syntax better and will use it in the discussion and examples in this article.

A third Prolog camp may group itself around Borland's implementation as Turbo Prolog uses a syntax different from either of the other two.

Procedural vs. Declarative Languages

The basic difference between Prolog and FORTRAN, BASIC, COBOL, Pascal, and the like is that Prolog is a declarative, or object-oriented, language while the others are procedural languages. A procedural language requires that the user explain the procedure to be used to solve a problem. A declarative language requires only that the problem be described.

In a declarative language you describe the goal and any constraints (starting point included); then the language does an exhaustive search to find a path to the answer. For example, suppose we want to sort a list. We might write a Pascal routine with a procedure header

```
Procedure SortList(X:BigArray;ArraySize:Integer;Var Y:BigArray)
```

where X is the input list and Y is the output list. In the procedure, we would have to explain how to sort. In a "bubble" sort, we would write code to do the following:

- One by one, examine all adjacent pairs of numbers. If a pair is such that the first number in the pair is larger than the second, swap them.
- Keep doing this until there are no more swaps to be done.

Here, we have described how to do what we want to have done. In Prolog, we would describe the conditions of the goal:

```
sort(X,Y) if permuted(X,Y) and inorder(Y) .
```

(This example was originally cooked up in 1974 by Robert A. Kowalski, the author of m-Prolog.)

In other words, the sort goal is achieved if there exists a permutation of X, called Y, which is arranged in order. Note that we must also define the goals permuted and in order.

Prolog basically proceeds by generating all possible permutations of the list X until it finds one that is in order.

For this reason, Prolog is not a particularly good language in which to write a sort. A 100 item list would require generating and testing 100 factorial or

93,326,215,443,944,152,681,699,238,856,266,700,490,
715,968,264,381,621,468,592,963,895,217,599,993,229,
915,608,941,463,976,156,518,286,253,697,920,827,223,
758,251,185,210,916,864,000,000,000,000,000,000,000,000

combinations.

On the other hand, Prolog code can be very compact, and it is a good language with which to prototype an idea. The idea can then be re-implemented in a declarative language.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Elements of Prolog: Atoms, Lists, Small Programs

Assertions in Prolog

The basic unit of Prolog is the atom and the predicate. It looks like

```
predicate(atom1,atom2....,atomN).
```

For instance, to express that Paul owns a Maserati, we could write

```
>owns(paul,maserati).
```

(The ">" is supplied by the computer as a prompt.)

Notice the use of lowercase: Names starting with uppercase are variables. The atoms "paul" and "maserati" are constants. Notice also the period: To end a statement in Prolog, use a period. Please understand that Prolog doesn't get any meaning out of this: It doesn't know what "own" or "paul" or "maserati" means. Atoms are symbols or words, but multiple words can be made one atom by surrounding them with quotes:

```
>loves("billy bob",beer).
```

Here, the name Billy Bob must be set off with quotes because it contains a space.

Queries in Prolog

To see what Paul owns, we can ask Prolog what it knows about Paul and ownership:

```
>owns(paul,X)?      <--- we type
  X = maserati      <--- it responds
```

Notice the use of the question mark and the uppercase letter. We are saying, "Your goal is to find a pattern that matches owns(paul,X), where X can be filled with anything." Lowercase letters for constants, uppercase for variables. Again, Prolog actually understands *none* of this. We could say,

```
>toves(slithy,mimsy).
```

and then ask,

```
>toves(slithy,X)?
```

and Prolog will reply,

```
"X = mimsy"
```

self-satisfiedly. By the way, I have just used two two-argument predicates: These aren't the only kind. I could have defined,

```
toves(slithy),
```

and asked,

```
"toves(X)?"
```

This information is now in the Prolog "database." Prolog maintains a database of facts. Many such facts can be in a database simultaneously:

```
>owns(john,skipper).
>owns(skipper,bone).
>owns(mary,house).
>owns(sue,house).
>owns(jack,hat).
```

and some others:

```
>loves(john,nap).
>loves(sue,house).
>loves(jack,mary).
>loves(jack,hat).
```

Then we could ask,

```
>owns(X,house)?
  X = mary
  X = sue
```

Notice that multiple solutions can be found. Similarly, if we had asked a more specific question, without variables:

```
>owns(jack,house)?
 NO
>owns(sue,house)?
 YES
```

Lists

Another Prolog data type is the list. Lists are designated with square brackets, and their items are separated with commas. [a,b,aardvark,"hum dinger"] is a typical list. In general, you can use lists just about anywhere simple atoms go:

```
>loves(jack,[racing,"pina coladas",music]).
```

A list is said to have a head and a tail. The head is the first list item. The tail is the list composed of all of the remaining list. A one item list [xx] has a head xx and a tail [ ], the "empty list." An alternative way of identifying the head and tail is the vertical bar:

```
>headof([Firstpart|Lastpart],Firstpart).
```

This is a Prolog rule that says, "The first argument should be a list, as you have described it as a list composed of a head and a tail. The second argument is an atom which is bound to (i.e., must be equal to) the head of the first argument." For example, querying this rule:

```
>headof([a,b,c],X)?
 X = a
```

Notice, by the way, that the variable Lastpart doesn't really do anything: Once something is bound to it, it is put to no use. It must, however, be there to indicate the list as a head and tail. In Prolog, we can express such a situation with the anonymous variable "_". This takes the place of a variable that is not used once bound. The above rule could be rewritten with the same result:

```
>headof([Firstpart|_],Firstpart).
```


PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Prolog tried to match [a,b,c] against the pattern of the first argument. The first argument looks like [H|T], a list divided into its head and its tail. It then bound Firstpart with the value a and "_" with [b,c], then looked at the second argument. If we had bound the second argument,

```
>headof([a,b,c],z)?
 NO
>headof([a,b,c],a)?
 YES
>headof([a,b,c],b)?
 NO
```

Writing Programs

Writing programs in Prolog essentially means defining logical rules that Prolog uses to manipulate its database. For instance, if John likes everything, we could define the rule

```
likes(john,_).
```

This is like a fact, but it's a general purpose fact. Such a fact is called a rule. It states that the "likes" goal is satisfied by a combination of John and anything (the "anonymous" variable). Notice that the "likes" rule has the same level of importance as "likes" facts. We can assert some universal truth, like "X is owned by Y if Y owns X," by using a rule. The above would be written

```
>is_owned_by(X,Y) :-  owns(Y,X) .
```

The ":- " is Prologian for "if." Note the use of variables: We have established a universal truth, which can now be queried:

```
>is_owned_by(A,B)?
 A = paul   B = maserati
 A = john   B = skipper
 A = skipper   B = bone
 A = mary    B = house
 A = sue     B = house
 A = jack    B = hat
```

That's all there is to writing Prolog programs: Just include some facts and rules and use a rule as the overall goal. Rules can also have multiple conditions. Suppose, for example, we want to define a rule bliss(X), where X has bliss if he/she loves something and owns it:

```
>bliss(X) :- owns(X,Y) , loves(X,Y) .
```

Note: The comma is Prologian for "AND." A rule is satisfied only if all clauses separated by commas are true.

Notice the use of the extra variable Y. If we query this rule:

```
>bliss(X)?
 X = sue
 X = jack
```

How, specifically, did Prolog do this? Basically, it just paws through the database to find all possible combinations that work. An imaginary trace might look like:

1. Hmmm. Well, let's start. What matches owns(X,Y)? X = paul, Y = maserati. Can I find a loves(paul,maserati)? (He'd better love it, for what he paid for it.) Nope, none there.
2. Now there's X = john, Y = skipper. Is there a loves(john,skipper)? Nope. . .
3. . . .continuing for a while. . .
4. Eventually, X = sue, Y = house. There is a loves(sue,house), so report success and continue as if you had failed.
5. . . .continue to the end. . . Finally, when the database is exhausted, stop.

This kind of search is called backtracking.

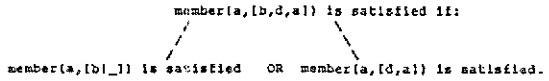
Suppose, however, that we must write a rule that involves looping. A common example is member(X,L), where member succeeds if X is a member of list L. How would we write this? Well, the FORTRAN programmer would reach for a DO loop, the BASIC programmer for a WHILE. . .WEND block, and the COBOL programmer for a PERFORM block. What do we have in Prolog? Nothing.

There are no looping constructs in Prolog due to its non-procedural nature. The only way to do something iteratively is to define it recursively. Recursion, recall, requires a stopping rule ("anchor") and an iteration rule (the recursion). The example of Member would be written in Prolog as two rules, the anchor and the iteration rule:

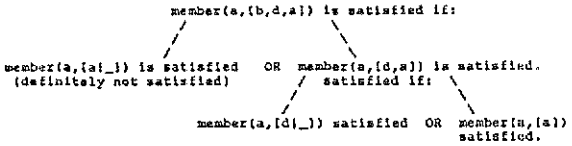
```
member(X,[X|_]).                  /* rule 1 -- the anchor */
member(X,[_|Y]) :- member(X,Y). /* rule 2 -- iteration rule */
```

The first rule says, "member succeeds if X is the head of the list." Again, the "_" is a Prolog construct that stands for a variable—any variable. We could have put A or B or something in there, but we didn't need it to bind to anything, so the general "_" was used. This is the anchor rule.

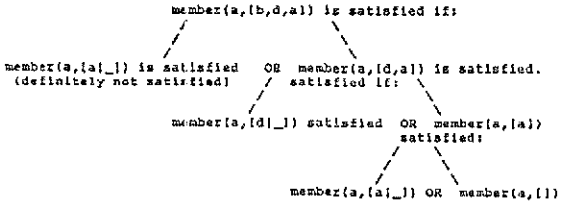
The second rule says, "member succeeds if X is a member of the tail of the list." This is the iteration rule. To see this in action, let's follow the execution of member(a,[b,d,a])? Prolog assembles its agenda like so:



Well, member(a,[b|_]) is definitely not satisfied. We must then try to solve the second. If we can't, the root goal is not satisfied.



Here, the tree is again expanded. member(a,[d|_]) is not satisfied, so we expand the OR tree:



Finally, we have a successful goal: member(a,[a|_]) matches. This propagates back up, and the root goal is solved.

Note that Prolog doesn't call one rule the anchor and the other the iteration rule; nor must there be only one of each.

Recursion can, however, be a large pain to have to work with. Referring back to our database, suppose I wanted to add the following rule:

If A owns B, and B owns C, then A owns C.

This would be "transitivity of ownership." How would I render this in Prolog? Simple, you may say—just add the following rule:

```
>owns(X,Y) :- owns(X,Z), owns(Z,Y) .
```

This would be called a "recursive" procedure because it refers to itself. (Some people say that "recursive" comes from "re" [to repeat something] and "curse" [to utter profanity], leading to "re-cursive": to curse again, which you do a lot of when doing recursive programming.)

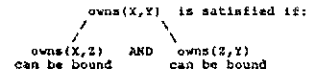
PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

But this won't work—if you ask,

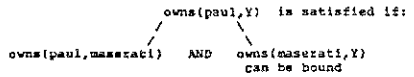
```
owns(X,Y)?
```

Prolog just thinks for a while, finally announcing something like "STACK OVERFLOW." Here's why:

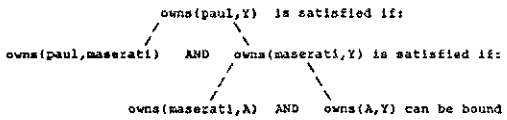
In order for owns(X,Y) to be satisfied, there must be either a fact that satisfies it or a rule allowing us to derive a new fact. For now, however, set aside the facts and look at how rules are satisfied. In this case, Prolog sets up a kind of "things to do" list, an AND/OR tree:



Having set that out, it sees what it can do. owns(paul, maserati) is known and can be both reported and used to try to fill out the tree:



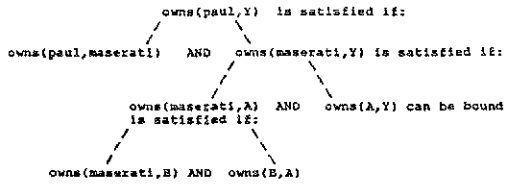
Now, Prolog asks, how can owns(maserati,Y) be satisfied? Well, as before, owns(maserati,Y) can be satisfied either by a fact in the database or by satisfying the two conditions in the "owns" rule. To see this, we must expand the tree, expanding the "things to do" list:



Thus, Prolog's "things to do" list now looks like:

1. Satisfy owns(maserati,A).
2. Use A to bind and satisfy owns(A,Y).
3. That will satisfy owns(maserati,Y) for the Y.
4. Once that's done, I will have satisfied owns(paul,Y) for the Y.

Now, to you and me, it is obvious that things aren't getting better, and having to satisfy owns(maserati,A) is no easier than satisfying owns(maserati,Z) was. Prolog, however, is by nature optimistic, trying every possible trick to satisfy this goal. It then says, "Well, let's dig into satisfying owns(maserati,A)," leading to


owns(paul,Y) is satisfied if:
owns(paul,maserati) AND owns(maserati,Y) is satisfied if:
owns(maserati,A) AND owns(A,Y) can be bound
is satisfied if:
owns(maserati,B) AND owns(B,A)

Prolog's "things to do" list now looks like

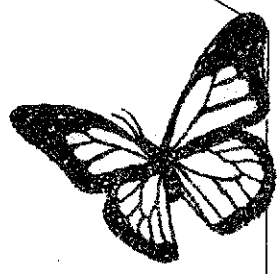
1. Satisfy owns(maserati,B).
2. Use B to bind and then satisfy owns(B,A) for some A.
3. Once A is bound, owns(maserati,A) is satisfied.
4. Use A to bind and satisfy owns(A,Y).
5. That will satisfy owns(maserati,Y) for the Y.
6. Once that's done, I will have satisfied owns(paul,Y) for the Y.

This process will continue until the space used for the "things to do" list—i.e., the stack—fills up. This is infinite recursion. Infinite recursion is a lot like standard infinite loops, but a little harder to detect and correct.

Chalcedony PROLOG $99.95

Not Copy Protected

A REAL Clocksin & Mellish Prolog for BOTH major microcomputing operating systems — with full cross-compatibility.



Complete with the predicates necessary for POWER AI programming:
op () name () functor () clause () =..("Univ")
...And no constraining data typing.

- Floating point
- Step-by-step tutorial
- Math functions
- Integrated editor

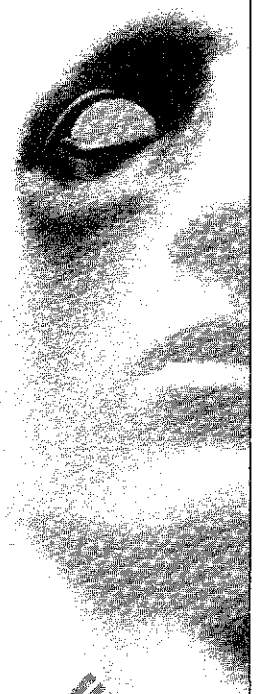
PROLOG/i

Extensible overlay library, 8087 support, large memory model (up to 640K)

PROLOG/m

Complete Macintosh environment with extensive pull-down menus and dialogue boxes.

No Risk Offer: Examine the PROLOG/i or PROLOG/m documentation at our risk for 30 days. If not completely satisfied, return with disk still sealed for refund.



APPLICATIONS—
Complete with SOURCE CODE

NFL X-pert $49.95
A true interactive expert system written by a professional knowledge engineer. A valuable learning tool for any Prolog programmer interested in using Prolog to develop expert systems.

TOOLBOX $29.95
More than 50 subroutines that speed and compress list handling, searches, sorts, and reversal algorithms. An inside look at the tricks of the professional Prolog programmer.

TOYBOX $29.95
Written by an academician to help his students understand Prolog, this collection of puzzles and mind-teasers will illustrate how the Prolog programmer creates programs that find the best solution to the problem. Turn your computer into a super reasoning machine!

CHALCEDONY SOFTWARE, INC.
5580 LA JOLLA BLVD.
SUITE 126
LA JOLLA, CA
92037
(619) 483-8513

System Requirements:
Minimum 256K RAM (320K recommended) PC DOS/MS-DOS ANSI Standard Support
Minimum 512K Macintosh Macintosh-plus and HFS Compatible

SAVE 10% when you buy either PROLOG/i or PROLOG/m and all 3 applications.

PHONE ORDERS: 1-800-621-0852 EXT 468

☐ PAYMENT ENCLOSED $ _____
CA residents add 6% sales tax
☐ CHARGE MY: ☐ MasterCard ☐ Visa
Card No. _____ Exp. Date _____
Signature _____
Mr./Mrs./Ms. _____
(please print full name)
Address _____
City/State/Zip _____

PROLOG/i	$99.95
PROLOG/m	99.95
Check:	
MS-DOS ☐	Mac ☐
NFL X-pert	49.95
TOOLBOX	29.95
TOYBOX	29.95
Complete Pack	188.82

SHIPPING:
$ 5.00 U.S.
7.50 Canada
10.00 Carribean, Hawaii Air
20.00 Overseas Air

Circle Number 79 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

ATTENTION TURBO PASCAL USERS!
Crash the 64K Barrier

Try TURBO PACKAGE now!
90 day money back guarantee!

Modular Programming!
Promotes REUSE of working CODE
CUTS development TIME
IMPROVES system RELIABILITY
SIMPLIFIES program MAINTENANCE

FILL 640KB with code/data any way you want
VERY FEW CODE CHANGES.
FASTER than chaining or overlaying

SUPERMATH, FREE!
With purchase of Turbo Package
40 plus LONG (32-bit math) routines
Faster than real - big enough for $.
ASM coding insures top performance

Just $49.95 (in TX add tax)
Visa/MC (no shipping chg)

Write or call for more information

CONVERSATIONAL COMPUTER SYSTEMS
5371 Verbena Rd.
San Antonio, TX 78240
Phone: (512) 692-0353

Circle Number 78 on the Reader Service Card

TURBO PASCAL GENERATOR

THE GTP PROFESSIONAL MODEL

Generate error-free Turbo Pascal source code for:

DATA BASES MENUS
SCREENS REPORTS

Build complete, working programs in minutes!

FEATURES
- Indexed Data Bases
- Multiple Screens
- Automatic Updating
- Built-in Edits
- Retrieval Facility
- Automatic File Build
- Quick Screen Handling
- Speedy DB Access
- Context Sensitive HELP
- Full Keyboard Support

EASY
1. Paint-the-screen
2. Define fields & calculations
3. Generate, compile & RUN

FLEXIBLE
- 100% Turbo Pascal
- Modify in Pascal under Turbo's Editor
- Extensions are easy with 130 page Programmer Reference manual
- 100 documented source code routines included

NOT Copy Protected NO Royalty Fees NO Run-time Library Required

Requires: IBM PC (100% compatible) 256K RAM 2 Disk Drives
PC-DOS 2.0+ Turbo Pascal 3.0 from Borland International

PRICE $200.00
CALL FOR QUANTITY DISCOUNTS
VISA/MC — Check — Money Order — No COD or Purchase Orders
Texas Residents: add $12.25 Sales Tax
Outside US & Canada: add $25.00 Air Postage and make payment by credit card or money order in US Funds.

AEF
ALLEN, EMERSON & FRANKLIN
P.O. Box 928
Katy, TX 77492
(713) 391-8570

Circle Number 53 on the Reader Service Card

The Cut and More on Backtracking

Sometimes, mere cleverness cannot be employed to keep a definition from going wild and infinitely recursing. Other times, we may not want all solutions, but just whichever solution ends up being the first to come along. To address this, Prolog has the cut. The cut is written "!" (exclamation point). The cut is somewhat like a "Halt" command in Pascal, or a "Goto" in FORTRAN or BASIC.

The cut is an instruction that says, "Don't backtrack past this point." To take a simple example, imagine the following database:

```
>person(paul).
>person(john).
>person(sue).
```

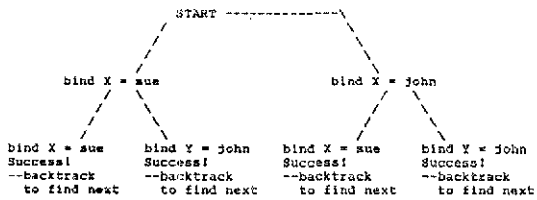
Then define a rule twopeople:

```
>twopeople(X,Y) :- person(X) , person(Y).
```

and query it:

```
>twopeople(X,Y)?
X = sue
Y = sue

X = sue
Y = john

X = john
Y = sue

X = john
Y = john
```

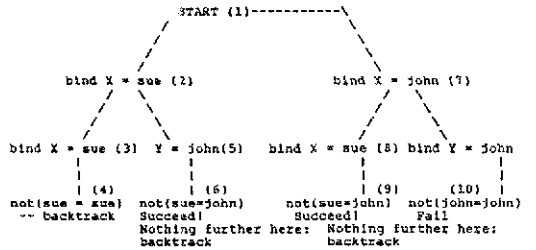
Prolog has found all possible answers to our question. It does this, as before, by constructing a tree of possibilities. The tree is generate-and-test using a depth-first search:


START
bind X = sue
bind X = john
bind X = sue Success! --backtrack to find next
bind Y = john Success! --backtrack to find next
bind X = sue Success! --backtrack to find next
bind Y = john Success! --backtrack to find next

Note Prolog's algorithm: It generates all possibilities, moving further down the tree so long as it hasn't failed yet. To make this clearer, take the following example. We might here define a rule disallowing duplicates:

```
>twoperson2(X,Y) :- person(X),person(Y),not(X=Y).
```

The new clause adds a level to the tree (numbers added to indicate order of actions):



Notice what happens when Prolog succeeds: In order to force discovery of all solutions, Prolog forces a "Fail" when a solution is found to force backtracking. If, however, we don't want Prolog to backtrack past a certain point, we can control this with the cut.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Look at the results of trying the cut at different locations:

```
>person(sue).
>person(john).
>t2(X,Y) :- person(X), person(Y) , !.
>t2(X,Y)?
X = sue Y = sue

>t3(X,Y) :- person(X), !, person(Y) .
>t3(X,Y)?
X = sue Y = sue
X = sue Y = john

>t4(X,Y) :-! , person(X), person(Y) .
>t4(X,Y)?
X = sue Y = sue
X = sue Y = john
X = john Y = sue
X = john Y = john
```

T2 introduced the cut at the end of the clause: This always means, "Show me only one solution, then stop."

T3 introduced the cut after person(X). This said, "Once you have successfully bound X, do not rebind." If we could not have succeeded with X = sue, there would have been no solutions reported. Thus, the two solutions with X = sue were reported.

T4 introduced the cut first, essentially at the root goal level. This does nothing as this is the default situation: Once all possibilities have been exhausted, the backtracking stops.

To underscore this, following are the same cuts with twoperson2, the rule with the extra "not(X=Y)" clause:

```
>person(sue).
>person(john).
>tp1(X, Y) :- person(X), person(Y), not(X = Y), !.
>tp2(X, Y) :- person(X), person(Y), !, not(X = Y).
>tp3(X, Y) :- person(X), !, person(Y), not(X = Y).
>tp4(X, Y) :- !, person(X), person(Y), not(X = Y).
```

Here, I define the four possible locations of the cut as four different rules.

```
>tp1(X,Y)?
X = sue Y = john
```

Here, as before, this essentially says "search until you find a success, then stop looking."

```
>tp2(X,Y)?
NO
```

This says, "Once you have bound X and Y, do not try to re-bind." Since the initial values of X and Y are X=sue, Y=sue, Prolog was committed to a nonsuccessful bind. The cut would not allow it to re-bind X or Y, so the overall goal failed.

```
>tp3(X,Y)?
X = sue Y = john
```

Here, Prolog is committed only to X=sue. It has enough flexibility to find Y=john. As before, an initial cut really doesn't do anything.

```
>tp4(X,Y)?
X = sue Y = john
X = john Y = sue
```

□

Mark Minasi is a partner at Moulton, Minasi & Company, a professional training and consulting firm with clients in the U.S., Canada, and western Europe. Moulton, Minasi & Company train in the fields of microcomputers, artificial intelligence, and data communications. Readers can reach Mark

c/o Moulton, Minasi & Company
5400 Wilson Blvd.
Arlington, VA 22205
(703) 276-0170

TURBO SOURCE SEARCH (TSS)

TSS is a **RBBS-by-mail** for **TURBO** Pascal. Stop paying AT&T, MCI, SPRINT et al. 10 times what it will cost you if you get it by mail from TSS.

Join **TSS** for $25. Then buy our library disks, **$2 per disk.** (45 disks and growing!!) Archived files = **700+ Kb** of code per **TSS** disk. **TSS** gives more bytes/buck! **Why pay more to get less?**

For your $25 you get
- 2 **FREE** library disks (utilities & catalog)
- 2 **FREE TSS** disks with your first order
- a validated password for the **online TSS-BBS**

TSS-BBS will be an "e-mail" drop-off for members to place orders then log off. It will also serve more traditional RBBS needs. Members will be able to DL the **TSS** disks **(at 10 times the cost of mails!)** if they really MUST have it now!

Join the **TSS BBS-thru-the-mail** for the best in public domain and shareware **TURBO** code.

TURBO SOURCE SEARCH
P.O. Box 876, Scituate, MA 02066, (617) 545-6677
payment: VISA/MC/COD if **validated**, otherwise check/money orders

Circle Number 35 on the Reader Service Card

If BASIC's data limit is the question . . .

DataSpace

. . . is the answer

DataSpace gives MS* and GW-BASIC* users . . .

- the ability to define and access string, integer, single, and double precision data arrays
- 10 - one, two or three-dimensional arrays up to 64K each - 640K total
- array storage and retrieval from files in any disc directory
- full compatibility with interpreters or compilers

. . . all with absolutely no impact on BASIC's own data space (requires MS-DOS* 2.10 or higher)

The only answer around for $75 . . .

DESIGN SOLUTIONS INC.
732 Sixth Avenue
Aurora, IL 60505
(312) 898-0651

* MS, GW-BASIC, and MS-DOS are trademarks of Microsoft Corporation

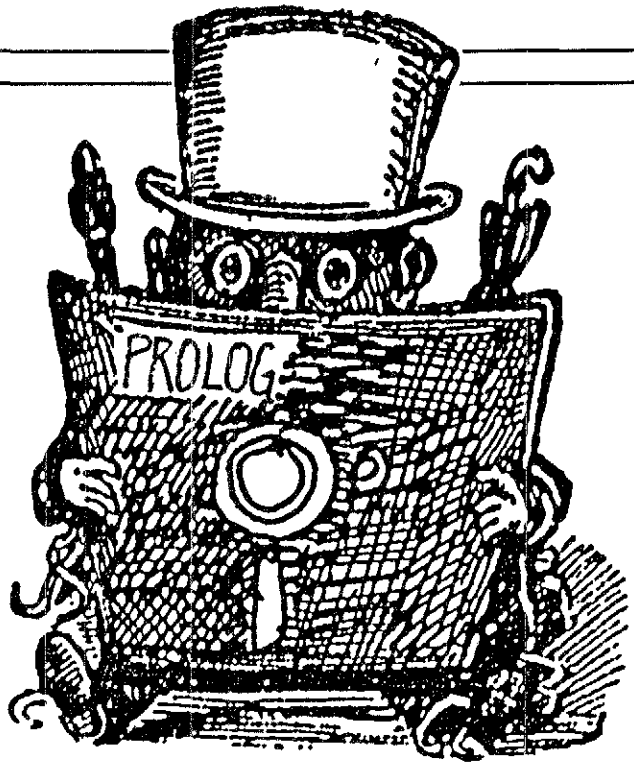
Circle Number 41 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

It's Not Real Prolog!

by Lew Baxter

A Prolog expert at Logicware, producers of MPROLOG, looks at Borland's Turbo Prolog.



Turbo Prolog is fast and slick—but it's not REAL Prolog!

First impressions are that it is a fast Prolog with a slick user interface, a slick manual, and plenty (66) of sample programs on the distribution disk (including a sophisticated natural language query system to a geographical data base). A peculiarity is that the Prolog is "typed"—the domains of predicates and structures must be declared in the program. This has a certain appeal from a software engineering viewpoint and at the same time undoubtedly allows for faster compilation techniques; however, the language now suffers from significant limitations. After a deeper analysis, it becomes apparent that Turbo-prolog is a *subset* of Prolog in two important areas. Both are conceptual limitations—it is virtually impossible to do "meta-programming" and any significant "dynamic" programming—both important requirements for serious AI programming.

Metaprogramming

One of the powers of Prolog (and LISP) is that programs and data have the same form, and therefore, data can be "executed" using the "evaluation" (a.k.a. "call") predicate. This, along with other "metaprogramming" predicates (e.g., "structure"), is absent from Turbo Prolog!

Dynamic programming

In Turbo Prolog you can dynamically manipulate *facts* but *not rules*!

Missing

Some significant MPROLOG features that Turbo-Prolog does not have are

Programming environment: interactive tracing; break facility.

Language: operators; Definite Clause Grammar.

Built-in predicates: exception handling; ancestor; evaluate; structure; string indexing; rich repertoire of dynamic handling predicates

Speed

Speed is 16500 lips on the AT, 4400 lips on the IBM PC XT (measured using naive reverse)—compared to 3650 lips on the VAX 750 and 260 lips (interpreter) on the IBM PC.

Space Requirements

Turbo Prolog needs 384K RAM.

User Interface

The user interface is slick, with fancy colored windows and pop-up menus. It features four system windows (which can be moved/enlarged):

Editor: likeWord Star; positions cursor at syntax errors
Dialog: user input and application output
Trace: single step, four-port spypoint; cursor is positioned in program also (must add "trace" compiler directive to program)
Message

LANGUAGE

DOMAINS: Specify domains; may get "domain error" at runtime

Standard domain types:
char e.g., 'a'
integer
real e.g., 12.34 1.2e123
string e.g., "John Smith"
symbol e.g., red "John Smith" (symbols and strings are interchangeable; handled differently internally)
file

Declare domains for predicates; e.g., append (list, list, list). Multiple declarations are allowed.
Declare domains for "compound objects"; e.g., sentence = sentence (noun, verbphrase); noun = noun (word); verbphrase = verbphrase (verb, noun); verb(word); verb = verb (word); word = symbol.
Repetitive domains; e.g., integer_list = integer
Recursive declarations; e.g., classlist = class (name, classlist); empty
Specify datatypes of arguments of predicates.
Statements use "if," "and," "or" (or ":- " " , " " . ").
In names, cases are preserved.
Integer range is −32768 to +32767.
Real arithmetic includes: sqrt, sin, cos, tan, arctan, exp, log.
"=" is used instead of "is."
Write allows an arbitrary number of args.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

COMPILER DIRECTIVES

- check mode usages
- check non-determinism
- suppress warnings
- trace
- set trail size

BUILT-IN PREDICATES

READING	
readchar (CHAR)	read a single char
readint (INT)	read an integer
readreal(REAL)	read a real
readterm(DOMAIN,TERM)	read a term
readln(STRING)	read a record
WRITING	
nl	
write(X1,X2,....Xn)	
writef(FORMAT,X1,....Xn)	formatted write
FILES	
DOS:	
deletefile(DOSFILE)	delete a DOS file
existfile(DOSFILE)	check if a DOS file exists
renamefile(OLD,NEW)	rename a DOS file
disk(PATH)	set/get DOS path
Open/close:	
openappend(FILE,DOSFILE)	open a file in append mode
openmodify(FILE,DOSFILE)	open a file for read and write
openread(FILE,DOSFILE)	open a file for reading
openwrite(FILE,DOS)	open a file for writing
closefile(FILE)	
readdevice(FILE)	set/get current read device
writedevice(FILE)	set/get current write device
Save/load DB:	
save(DOSFILE)	save DB into a DOS file
consult (DOSFILE)	read DOS file into DB
Others:	
dir (PATH,FILESPEC,DOSFILE)	select a file using the directory window
eof(FILE)	check for eof
filepos(FILE,POS,MODE)	set/get file position
file_str(DOSFILE,STRING)	read an entire file
flush(FILE)	flush file buffer
SCREEN	
attribute(X)	set/get screen attributes
graphics(MODE,PAL,COLOR)	initialise screen
text	reset screen to text mode
Turtle graphics:	
back(STEP)	move back
forward(STEP)	move forward
left(ANGLE)	set/get direction
right(ANGLE)	set/get direction
pencolour (COLOR)	set pen color
pendown	put pen down
penup	put pen up
line(R1,C1,R2,C2,COLOR)	draw a line
Attributes:	
scr_attr(ROW,COL,ATTR)	set/get char attributes
field_attr(ROW,COL,LEN,ATTR)	set/get attributes of a field
window_attr(ATTR)	set attribute of current window
Window:	
clearwindow	clear currently active window
makewindow(...)	define a window with attributes
removewindow	remove window from screen
shiftwindow(WINDOW)	set/get current window
edit(IN,OUT)	use editor
editmsg(IN,OUT,...)	use editor with options
I/O:	
scr_char(ROW,COL,CHAR)	read/write into a char
field_str(ROW,COL,LEN,STR)	read/write into a field
window_str(STRING)	read/write entire window
Cursor:	
cursor (ROW,COL)	set/get cursor
cursorform(START,END)	set cursor height
display(STRING)	display a string in window
dot(ROW,COL,COLOR)	set/get color of a pixel
STRING	
concat(STR,STR,STR)	concatenation (multipurpose)
frontchar(STR,CHAR,REST)	split string into 1st and rest
frontstr(N,STR,START,REST)	split string into N chars and rest
fronttoken(S,TOKEN,REST)	split string into token and rest
isname(X)	check for name
str_len(STRING,LENGTH)	string length
TYPE CONVERSION	
char_int(CHAR,CODE)	convert char into ASCII code
str_char(STR,CHAR)	convert between string and char
str_int(STRING,INT)	convert between string and integer
str_real(STRING,REAL)	convert between string and real
upper_lower(S1,S2)	convert between cases
DATABASE	
asserta(FACT)	add a fact to DB
assertz(FACT)	add a fact to DB
retract(FACT)	delete first matching fact in DB
SYSTEM	
beep	beep speaker
sound(DUR,FREQ)	play a note
bios(INT,REGS1,REGS2)	cause BIOS interrupt
date(Y,M,D)	set/get date
time(HOUR,M,S,H)	set/get time
membyte(SEG,OFFSET,BYTE)	set/get memory byte
memword(SEG,OFFSET,WORD)	set/get memory word
portbyte(PORT,X)	send/receive I/O port
ptr_dword(STR,SEG,OFFSET)	set/get memory address
storage(STACK,HEAP,TRAIL)	get Prolog memory sizes
system(DOSCOMMAND)	execute DOS command
trace(STATUS)	set/get tracing to on/off
MISC	
free	check if not bound
bound	check if bound
fail	
exit	stop and return to System
find all(VAR,ATOM,LIST)	'set_of'
not	
random(REAL)	

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

TURBO PROLOG PROGRAMMING SYSTEM

RUN	Execute program in memory—either program has a goal or the user interactively enters it.
COMPILE	Compile and produce program in memory, in an OBJ file, or in an EXE file (with automatice linking can link using DOS commands).
OPTIONS	Specify compilation options.
EDIT	Edit workfile.
FILES	Load—from a DOS file Save—save current workfile into a DOS file Dir—select default directory and browse files File—rename workfile Zap—erase text in editor Print—program on printer Erase—a disk file Rename—a disk file Os—call DOS
SETUP	Define directories for PRO OBJ EXE TURBO DOS Librarian—manage modules in a "project" Window—modify position and size of System Windows: Edit, Dialog, Message and Trace windows Color—set fore/back colors of System Windows, Pull Down menus and Status Line Misc—set stack size Save—current setup
QUIT	Leave Turbo Prolog and save workfile if necessary

GENERATING STAND-ALONE PROGRAMS

Compile and link modules separately.
A "project" is a collection of modules.
Global declarations specify inter module communications.

EXTERNAL LANGUAGE INTERFACE

To Pascal, C, Fortran, and Assembler
Pass integer, real, char, string, symbol, and compound

Significant MPROLOG features *not* present in TURBO PROLOG

hierarchical modularity
exception handling
operators
large range of integers
cross reference dictionaries
portability
interactive tracing
break facility
DCG
string indexing
metaprogramming predicates: evaluate, structure, ancestor
no "sort" predicate or "@<" (generalised comparison)

Additional comments from User Nets, etc. . . .

Turbo Prolog lists are too specific. Constructs such as

[1, [dog,cat], [1, 2, 3], ["string"]]

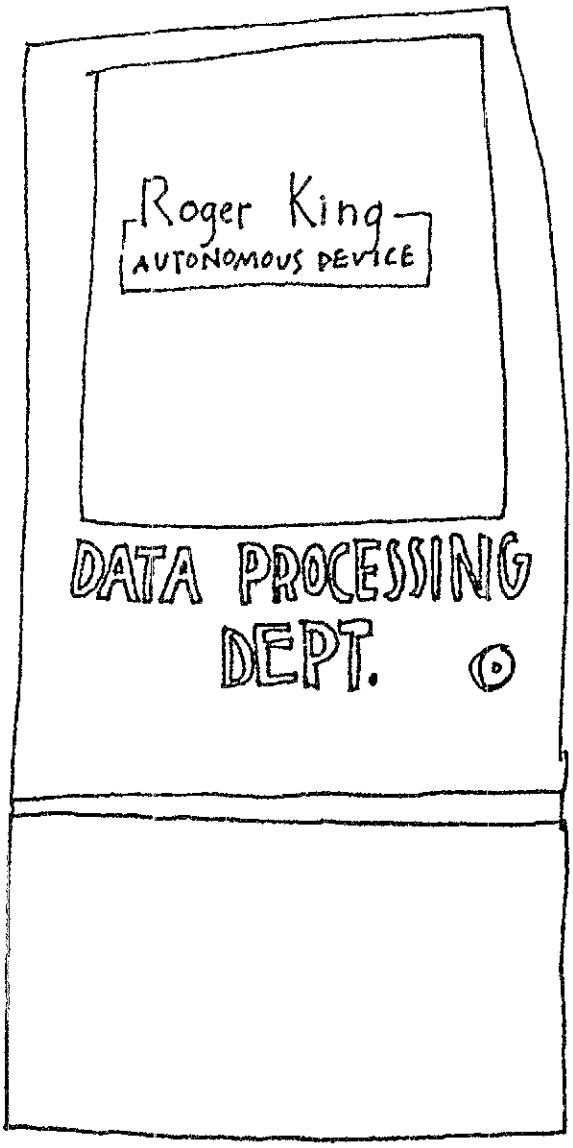
aren't legal because lists must be made of items of the same type (i.e., integers, reals, symbols, etc.). Plus, even if the list were modified as follows,

[1, [10, 20], [1, 2, 3], [999]]

it would be legal only if its exact shape were declared previously.

Conclusion

I think Turbo Prolog will do a lot to stimulate sales of other, more complete, competing Prologs. Arity, who make an expensive but fairly complete Prolog implementation for the PC, claim that they're overjoyed by Turbo Prolog and foresee increased sales of their product due solely to people's getting exposed to Prolog and then dissatisfied by Borland's offering. □



PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Turbo Prolog: Another Miracle?

by Tom Swan

Borland's bhagwan has done it again. Repeating the tenets—low price, simple packaging, high speed—that led to Turbo Pascal's apotheosis, Philippe Kahn has delivered another miracle. Turbo Prolog has come. And the word is good.

Or is it? Some say no. As many say yes. With rumblings of witch burnings and guru gashing, this is what they say. It's not Prolog, they say. But it's fast and we like it, they say. You'll be sorry, they say. No we won't, they say. It doesn't follow Clocksin and Mellish, they say. Neither do the others, they say. You can't program metalogically, they say. Who cares, they say, we don't even know what that is, and besides, it has windows. What have windows got to do with real Prolog, they say. What has real Prolog got to do with getting our jobs done, they say.

Who's right? That's a tough question and I don't have a simple answer. Because Prolog, an abbreviation of "Programming in Logic," is not my usual cup of holy water, I'll avoid the theological debate and concentrate instead on three points. Is Turbo Prolog worth the money? What's it good for? Will it replace Pascal?

Is It Worth the Money?

The price disappointed me at first. I had hoped that Turbo Prolog would sell for $49.95, the same bargain basement figure that made Borland famous for low priced, quality software. I remember thinking when I first heard of Turbo Pascal, "I can afford $49.95 just to see what it's all about." But recently, when I saw the Turbo Prolog ads, I thought, "Here we go again, another whatever-the-market-will-bear pricing strategy."

I was wrong. If you consider that $99 buys not only a fast native-code compiler, but also a window manager, a WordStar-like editor that you can call from programs, a relational database, plus a DOS directory file system, you will realize—as I do now—that Turbo Prolog is easily half if not one-quarter the price of any equally capable development system on the market.

So yes, it's worth $99. One small part of me still wishes for that old $49.95 number—and a big part of me hopes that the higher price is not a trend. (Will Modula-2 come out at $149.95? I hope not.) But I'm satisfied that Turbo Prolog is fairly priced. You get more than you pay for.

What Do You Get?

Turbo Prolog comes with a 221-page manual and two IBM PC DOS double-sided diskettes. The manual, whose cover *BYTE* artist Robert Tinney designed, is adequate, and the software is simple to install. Before trying the software, though, run the README program for a list of corrections and additions. (Hard disk users must also run INSTALL, apparently to tell Prolog to use drive C: as the default drive.)

The manual contains a few seriously misleading errors. For example, sample programs that list directories inside windows do not work and all of the graphics examples need changes before they draw anything on my standard IBM Graphics Adapter and color monitor. Also, Step 1 on page 154 says, "Create a LIBRARIAN file via the Librarian entry in Setup. . . ." This should read, "Create a PROJECT file via the Module List entry in Files. . . ." I could find no other references to the ephemeral Librarian. And there are other problems. Several commands are missing from the standard predicates directory in the appendix, and one, Pencolour, is misspelled Pencolor—proof of the European origins of this compiler.

I'm irked also by an overabundance of hyphens. Reconfigure, misspelling, and reassign are not, for heaven's sake, hyphenated words. I'm a picky writer and such things bother me more than they probably bother most people. But I give the manual only average marks on editing. It seems to me that Borland hired one of Computerdom's top artists to do the cover. They should have hired a top writer to do the insides too. [After this review was written, Borland announced that a revised version of Turbo Prolog will be released, including changes in the manual.—Ed.]

In its favor, the manual's tutorial has everything you need to start using Prolog and learn what it can do. The many examples are good, and on disk you'll find still more sample Prolog programs. HANOI.PRO is cute. GEOBASE is an impressive, though limited, almanac-type database. But to learn Prolog, you'll need a language reference—W.F. Clocksin and C.S. Mellish's *Programming in Prolog*, Springer-Verlag, Second Edition, 1984, is a good place to start. Written while the authors were employed by the Department of Artificial Intelligence at the University of Edinburgh, Scotland, the book describes what has become known as the "Edinburgh" Prolog standard. With that book and the Turbo manual, you have enough material to keep you busy for weeks and to start writing your own programs.

What's It Good For?

I'm amazed to hear me, confirmed Pascal pundit that I am, say it, but programming in Turbo Prolog is a joy. With a few simple lines, you can do some wonderfully complex things. For example, although it's not a Prolog standard operation, one of the neatest things you can do is add a WordStar-like editor to your own programs. To demonstrate, I wrote the following program, which I call PROSTAR.

```
/*
 * PROGRAM: ProStar text editor
 * SYSTEM:  Turbo Prolog 1.00
 * AUTHOR:  Tom Swan
 *
 */

PREDICATES
        ProStar
GOAL
        ProStar.
CLAUSES
        ProStar:-
                makewindow( 1, 7, 16, "ProStar Editor", 0, 0, 24, 80 ),
                edit("",_).
```

That also shows the form most Prolog programs take. The PREDICATES section describes the form of later clauses. Here, I'm simply declaring that I intend to use a clause named prostar. The GOAL is what the program attempts to solve. In this case, Prolog tries to prove that prostar is true, and in the process of doing that, runs the program. Finally, the CLAUSES section describes to Prolog how to satisfy its goal. Missing from this

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

example is a DOMAINS section in which you declare new data types.

Not only does this simple program duplicate the entire Prolog text editor, but it also gives you a 64K editing space, windowed help screens, DOS directories, the usual array of editing commands, and a secondary edit window in which you can temporarily edit another file without leaving the main one. There's no question that Turbo Prolog has one of the most powerful user interfaces ever seen in a software development system. And you don't pay any royalties to use it. What other compiler lets you add WordStar to your programs with a single command at no extra charge?

You can also directly access DOS in several ways. One way is to use the predicate "system" to execute a DOS command as a string. For example, to list a directory of all the .PRO files on the current drive, you could write commands such as the following. Then, to get back to Prolog from DOS, type EXIT.

```
system( "DIR *.PRO" ).
system( "TURBO" ).
system( "" ).
```

Borland confirmed my suspicion that the system command executes a fresh copy of COMMAND.COM as a child process running on top of Prolog. Because of that, Prolog (the parent process) remains in memory while you execute DOS commands. According to the manual, the command is supposed to operate inside windows, but that does not work properly. I'm told this acknowledged bug will not soon be fixed. If you want directories in windows, you have to use the Dir predicate instead.

Prolog is memory hungry. You need at least 384K of RAM to run it. You can directly compile and run programs or compile them to stand-alone .EXE files. The results take a lot of room, though. In one test, a bare minimum program with a single write clause took 45,983 bytes in .EXE form. You can also create separate .OBJ files that you can link to other Prolog modules or to those written in C or assembly language. Successful linking requires Version 2.2 or later of LINK.COM, which comes with Microsoft's Macro assembler package. The linker supplied with DOS 2.10 will not work. But even with the correct linker, you cannot link Prolog and Turbo Pascal programs because that language doesn't generate .OBJ files. Borland promises in the back of the manual to repair this slight with the release in 1987 of Turbo Pascal 4.0.

Data types include the usual real, integer, char, and string, along with symbol (similar to enumerated types in Pascal but interchangeable with strings) and file. Prolog stores reals in IEEE eight-byte format for a range of +/-1e-307 to +/-1e+308. Integers and reals can be mixed in expressions. Strings can be any length up to 64K. You add comments by surrounding them with /* and */. The manual doesn't say so, but comments can be nested. I don't know the maximum depth, but I successfully nested comments up to 14 levels deep.

Four main windows, Edit, Dialog, Trace, and Message, take up the entire display. Their arrangement is adequate for demonstrations but not so good for developing long programs. Don't be concerned, though. Adjustments are easy to make. One of the first things I did was to shrink the Trace and Message windows to about four lines each and move the Edit window on top of Dialog. While editing, I see the Edit window. But when running programs, I see the Dialog window in its place. The manual doesn't explain that windows can overlap that way, but it works.

The Trace window comes alive when you add "trace" or "shorttrace" to the top of a program. These debugging commands let you view every Prolog call and return. I used them on simple examples to learn how Prolog goes about its job. Tracing programs is a great way to quickly learn what Prolog can and cannot do.

I could go on and on about various extensions and operating system commands. Turbo Prolog is rich with features not available in any other Prolog system. There's a full array of window, string, and DOS related commands to display directories, delete files, and change subdirectories. There are commands such as asserta and assertz for creating databases and consult and save for reading and writing database files to disk. On the negative side, graphics are skimpy and slow (a simple test took 17 seconds to draw a circle with a diameter of about ¼ display size). Borland tells me they have a toolbox under development that will improve Prolog's graphics. That's good news. It's needed.

But Is It Prolog?

Peter Gable, president of Arity Corporation, and Lew Baxter of Logicware spent a half hour on the phone each explaining to me why they think Turbo Prolog is not really Prolog. Of course, you would expect competitors to be critical of their competition, but these two Prolog experts raised three possibly valid objections: the addition of type checking; the inability to program metalogically; and the requirement that lists can have only objects of the same type. Let's take their criticisms one at a time.

In most Prologs, you can freely interchange objects of different types. As in BASIC, when you want to use a new variable, you just start using it. In Turbo Prolog, however, you have to define new data types—called DOMAINS—before you can use them. Furthermore, you have to declare the form of clauses (similar to procedures and functions in Pascal) in a nonstandard PREDICATES section. While these requirements do make Turbo Prolog different from the standard, I'm not certain they're bad ideas. It's well known that languages with no type checking encourage poor programming. It's true also that type checking is not in the Edinburgh standard. I'll take a stand and vote in favor of Turbo Prolog on this one. Anything that helps me write correct programs is welcome.

Their second objection, the lack of metalogical programming, is valid. In simple terms, a metalogical program is one that modifies itself while running, an aspect of Prolog programming that Artificial Intelligence researchers see as critical to their work. I was going to tell you that Borland goofed by eliminating this important Prolog feature, but I just learned that a planned upgrade to Version 1.02 will bring the compiler closer to the Clocksin and Mellish standard. I was told the upgrade might be available by the time you read this review. If not, it certainly will be ready within weeks. Borland has heard the screams from the AI crowd and plans to respond.

Finally, the criticism that you cannot have lists of different object types in Turbo Prolog is only partially true. The following program shows one solution to the problem.

```
/*
 * PROGRAM : ListDemo
 * SYSTEM  : Turbo Prolog 1.00
 * AUTHOR  : Tom Swan
 *
 */

DOMAINS
        object = int(integer) ; str(string) ; real(real)
        list = object*
```

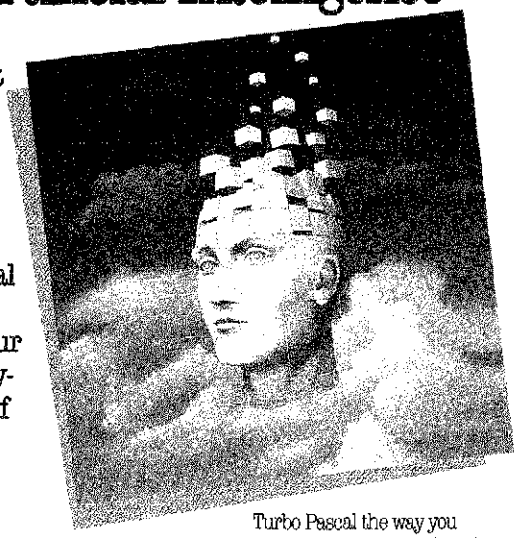

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Step-by-step tutorial, demo programs with source code included!

Borland's new Turbo Prolog is the powerful, completely natural introduction to Artificial Intelligence

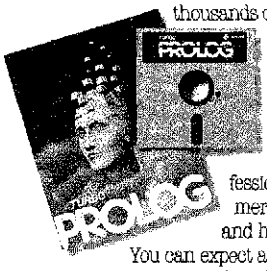
Prolog is probably one of the most powerful computer programming languages ever conceived, which is why we've made it our *second* language—and "turbocharged" it to create Turbo Prolog.™

Our new Turbo Prolog, the natural language of **A**rtificial **I**ntelligence, brings supercomputer power to your IBM® PC and introduces you step-by-step to the fascinating new world of **A**rtificial **I**ntelligence. And does all this for an astounding $99.95.



Turbo Prolog is to Prolog what Turbo Pascal® is to Pascal!

Our Turbo Pascal astonished everyone who thought of Pascal as "just another language." We changed all that—and now Turbo Pascal is the de facto worldwide standard, with hundreds of thousands of enthusiasts and users in universities, research centers, schools, and with professional programmers, students, and hobbyists.



You can expect at least the same impact from Turbo Prolog, because while Turbo Prolog is the most revolutionary and natural programming language, it is also a complete development environment—just like Turbo Pascal.

> "Turbo Prolog offers generally the fastest and most approachable implementation of Prolog."
>
> Darryl Rubin, AI Expert

Even if you've never programmed before, our free tutorial will get you started right away

You'll get started right away because we have included a complete step-by-step tutorial as part of the 200-page Turbo Prolog Reference Manual. Our tutorial will take you by the hand and teach you everything you're likely to need to know about Turbo Prolog and artificial intelligence.

For example: once you've completed the tutorial, you'll be able to design your own expert systems utilizing Turbo Prolog's powerful problem-solving capabilities.

Think of Turbo Prolog as a high-speed electronic detective. First you feed it information and teach it rules. Then Turbo Prolog "thinks" the problem through and comes up with all the reasonable answers—almost instantly.

If you think that this is amazing, you just need to remember that Turbo Prolog is a 5th-generation language—and the kind of language that 21st century computers will use routinely. In fact, you can compare Turbo Prolog to Turbo Pascal the way you could compare Turbo Pascal to machine language.

You get the complete Turbo Prolog programming system for only $99.95

You get a complete Turbo Prolog development system including:

- The lightning-fast Turbo Prolog incremental compiler and the interactive Turbo Prolog editor.
- The 200-page reference manual which includes the step-by-step Turbo Prolog tutorial.
- The *free* GeoBase™ natural query language database including commented source code on disk—ready to compile. GeoBase is a complete database designed and developed around U.S. geography. It includes cities, mountains, rivers, and highways, and comes complete with natural query language. Use GeoBase immediately "as is," or modify it to fit your own interests.

So don't delay—don't waste a second—get Turbo Prolog now. $99.95 is an amazingly small price to pay to become an immediate authority—an instant expert on artificial intelligence! The 21st century is only one phone call away.

Turbo Prolog 1.0 Technical Specifications

Programming System Features

- ☑ ***Compiler:*** Incremental compiler generating native in-line code and linkable object modules. The linking format includes a linker and is compatible with the PC-DOS linker. Large memory model support. Compiles over 2500 lines per minute on a standard IBM PC.
- ☑ ***Interactive Editor:*** The system includes a powerful interactive full-screen text editor. If the compiler detects an error, the editor automatically positions the cursor appropriately in the source code. At run-time, Turbo Prolog programs can call the editor, and view the running program's source code.
- ☑ ***Type System:*** A flexible object-oriented type system is supported.
- ☑ ***Windowing Support:*** The system supports both graphic and text windows.
- ☑ ***Input/Output:*** Full I/O facilities, including formatted I/O, streams, and random access files.
- ☑ ***Numeric Ranges:*** Integers: −32767 to 32767; Reals: 1E−307 to 1E+308
- ☑ ***Debugging:*** Complete built-in trace debugging capabilities allowing single stepping of programs.

YES! I want the best

Turbo Prolog at only:

$99.95

To order by phone, or for a dealer nearest you,

Call (800) 255-8008

in CA call (800) 742-1133.

Send me ____ Turbo Prolog at $ ____

Outside USA add $10 per copy
CA and MA res. add applicable sales tax $ ____

Amount enclosed. $ ____

This price includes shipping to all US cities

Payment: VISA MC Bank Draft Check

Credit card expiration date: ___/___

Card #

†You must have an IBM or true compatible running DOS 2.0 or later.
My computer's name and model is:

The disk size I use is: ☐ 3½" ☐ 5¼"

NOT COPY PROTECTED
****60-DAY MONEY-BACK GUARANTEE***

Name:

Shipping Address:

City:

State: *Zip:*

Telephone:

CODs and purchase orders WILL NOT be accepted by Borland. Outside USA make payment by bank draft payable in US dollars drawn on a US bank.

**YES, if within 60 days of purchase this product does not perform in accordance with our claims, please call our customer service department and we will gladly arrange a refund.*

†Minimum system requirements:
IBM PC, XT, AT, PCjr, and true compatibles; 384K RAM.

SU36

BORLAND INTERNATIONAL
4585 SCOTTS VALLEY DRIVE
SCOTTS VALLEY, CA 95066
(408) 438-8400 TELEX 172373

Vive la différence

Borland products include Turbo Prolog, Turbo Pascal, Turbo Tutor, Turbo Editor Toolbar, Turbo Database Toolbox, Turbo Graphix Toolbox, Turbo GameWorks, Turbo Lightning, Lightning Word Wizard, Reflex, The Analyst, Reflex Workshop, SideKick, SideKick, The Macintosh Office Manager, Travelling SideKick, and SuperKey — all of which are trademarks or registered trademarks of Borland International, Inc. or Borland/Analytica, Inc.
Copyright ©1986 Borland International 8-10495E

IBM, XT, AT, and PCjr are registered trademarks of International Business Machines Corp.

Circle Number 3 on the Reader Service Card.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

FORTRAN PROGRAMMERS

Looking for the right PC FORTRAN LANGUAGE SYSTEM? If you're serious about your FORTRAN programming then you should be using F77L- LAHEY FORTRAN.

Editor's Choice - PC Magazine

- Full FORTRAN 77 Standard (F77L is not a subset)
- Popular Extensions for easy porting of minicomputer and mainframe applications
- COMPLEX*16, LOGICAL*1 and INTEGER*2
- Recursion - allocates local variables on the stack
- IEEE - Standard Floating Point Arithmetic
- IMPLICIT NONE
- Long variable names - 31 characters
- Fast Compile - Increase your productivity
- Source on Line Debugger (Advanced features without recompiling)
- Arrays and Common Blocks greater than 64K
- Clear and Precise English Diagnostics
- Compatibility with Popular 3rd Party Software (i.e. Lattice C)
- Easy to use manual
- Technical Support from LCS

F77L - THE PROGRAMMER'S FORTRAN

$477.00 U.S.

System Requirements: MS-DOS or PC-DOS, 256K, math coprocessor (8087/80287)

FOR MORE INFORMATION: (702) 831-2500

Lahey Computer Systems Inc.
P.O. Box 6091 Incline Village, NV 89450/U.S.A.

International Dealers:

England:	Grey Matter Ltd.,	Tel: (0364) 53499
Denmark:	Ravenholm Computing,	Tel: (02) 887249
Australia:	Computer Transitions	Tel: (03) 537-2786
Japan:	Microsoftware, Inc.,	Tel: (03) 813-8222

SERVING THE FORTRAN COMMUNITY SINCE 1967

Circle Number 44 on the Reader Service Card

Sort RM/COBOL Files

• • • **QISORTC** • • •

Finally, a sort utility for RM/COBOL. Using efficient algorithms in assembly language making it the fastest sort on the MS-DOS market. Qisortc supports:

- RM/COBOL Version 1.5x & 2.xx files
  - Single / Dual File Indexed
  - Relative and Sequential
- ASCII, COMP-1, & COMP-3 fields
- 32 sort control fields
  - ascending / descending
- 32 record selection fields
  - EQ:NE:GT:GE:LT:LE field:constant
  - implied AND, OR may be specified
- 32 output fields or total record
- CALL from RM/COBOL, when provided subroutine is linked with runtime
- Use pathnames and return codes
- Keyboard or Parameter file
- Requires MS-DOS ver 2.xx or above
- Only $95.00 - COD, VISA, or M/C.

Quantumn Information Systems, Inc.
145 N.W. 85th St., Suite 103
Seattle, Washington 98117
(206) 789-2888 Telex 4941041

Circle Number 31 on the Reader Service Card


```
PREDICATES
        member(object,list).

CLAUSES
        member(X,[X|_]).
        member(X,[_|L]):-
                member(X,L).
```

LISTDEMO is a modified version of a common Prolog algorithm that determines whether an object is a member of a list. Lists are enclosed in brackets and can have any number of elements. For example, [12,145,99] is a list of three integers. The usual member predicate checks for the presence of one member in a list. If you type the goal, member (145,[12,145, 99]), Prolog reports True because 145 is a member of the list. To solve this problem, the member predicate uses the | operator to separate the head of a list from its tail. By doing this over and over, eventually it either finds the hunted element or ends with an empty list.

But how do we do that with lists of different kinds of objects—strings, real numbers, characters, and so on? My answer is to create an object whose type is a variety of *functors*—a funny word that refers to structures such as int(integer), str(string), and reals(real). LISTDEMO declares object this way, separating the multiple functors with semicolons.

Now that we've done that, instead of creating lists of reals, strings, and integers, we can instead construct lists of multiproperty objects. (The * in object* creates a list of objects.) Running LISTDEMO and typing the goal:

```
member(int(5),[real(3.14159),str("Abcd"),int(5)]).
```

returns "True" because int(5) is a member of the object list. This list contains a real number, a string, and an integer—all represented as objects. But there's nothing magical happening. The functors real( ), str( ), and int( ) are just pointers. And the list is just a list of those pointers. The program manually does what I suspect standard Prolog does internally to maintain lists of different data types. While this approach works, I'm curious whether Prolog experts will accept my solution. (I'm sure I'll find out soon enough!)

While I admit to a limited knowledge of Prolog, I think the various criticisms are less important than many think. Still, the conclusion is unavoidable. Turbo Prolog 1.00 is not standard Edinburgh Prolog. The upcoming Turbo 1.02 won't be standard either, but it will be much closer. Whether these differences are improvements or detractions is not so easy to tell. It's a story I plan to watch closely.

Programming in Logic

Writing Prolog programs is as simple as stating a few facts and rules. That's all a Prolog program is—a collection of facts and a list of rules that describe true and false things about those facts. Although statement order is important in sequential languages such as BASIC and Pascal, it plays only a minor role in Prolog. To write a program, you just start telling it the things that you know to be true.

The fun begins when you give Prolog a goal, which you can do in one of two ways. If you directly type goals into a program, as in the earlier PROSTAR example, Prolog automatically attempts to evaluate them. If you don't supply an explicit goal, as in LISTDEMO, Prolog asks you to type one in its Dialog window. Either way, Prolog attempts to evaluate goals according to the facts and rules it knows.

The difference between that and a conventional program in Pascal or Modula-2 is easier to see in an example, the idea for which I take in part from D.E. Cortesi's excellent article, "Tour of Prolog" (*Dr. Dobb's Journal* [March 1985], 44–63). If you

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

have Turbo Prolog, type in the following program. If not, skip to the discussion that follows. You don't need to run the program to understand what it does.

```
/*
 * PROGRAM  : SUM predicate demonstration
 * SYSTEM   : Turbo Prolog 1.00
 * AUTHOR   : Tom Swan
 *
 */

PREDICATES
        sum(integer,integer,integer)
        succ(integer,integer)
        pred(integer,integer)
        randint(integer)

CLAUSES

/* one free variable */

        sum(X,Y,Z):- bound(X),bound(Y),Z=X+Y.
        sum(X,Y,Z):- bound(Y),bound(Z),X=Z-Y.
        sum(X,Y,Z):- bound(X),bound(Z),Y=Z-X.


/* two free variables */

        sum(X,Y,Z):- free(X),free(Y),bound(Z),randint(X),Y=Z-X.
        sum(X,Y,Z):- free(X),free(Z),bound(Y),randint(X),Z=X+Y.
        sum(X,Y,Z):- free(Y),free(Z),bound(X),randint(Y),Z=X+Y.


/* three free variables */

        sum(X,Y,Z):- free(X),free(Y),free(Z),randint(X),randint(Y),Z=X+Y.


/* integer random number generator */

        randint(X):- random(Y),X=Y*32767.0.


/* successor relation */

        succ(X,Y):-
                sum(1,X,Y).


/* predecessor relation */

        pred(X,Y):-
                succ(Y,X).
```

Parts of SUM come from an example in the Prolog manual, but I modified the example to fulfill Cortesi's wish for fully descriptive arithmetic in micro-Prolog, the compiler he used in the article. To understand the idea of descriptive arithmetic, consider how Prolog evaluates the sum predicate. For example, we want to type something like this:

```
sum( 15, 24, Result ).
```

and have Prolog report that Result = 39. Although it may seem that Prolog merely added 15 + 24 to calculate the Result, it logically finds Result as though it looked it up in a database of true statements such as these:

```
sum( 0, 0, 0 ).
sum( 1, 0, 1 ).
sum( 0, 1, 1 ).
sum( 1, 1, 2 ).
   :
sum( 15, 24, 39 ).
   :
sum( 99, 100, 199 ).
```

Prolog simply finds the known quantities and reports the unknowns. Because of that, you can type the goal:

```
sum( Result, 24, 39 ).
```

and Prolog reports Result = 15. In other words, you can subtract by using the sum predicate in unplanned ways! Remember—at least logically speaking—Prolog looks for a true clause where 24 and 39 appear in the second and third positions. It then fills in the unknown Result. If you ask for more than one unknown this way:

```
sum( A, B, 40 ).
```

Prolog finds two numbers that add up to 40. You can also specify goals such as sum(A, 25, C), or ask for three unknowns

PRODUCTIVITY TOOLS

From Opt-Tech Data Processing

NEW Version

Opt-Tech Sort™

NEW Version

ALL NEW Version 3.0 features even faster sorting, record selection, output record reformatting, dBASE III files, comma delimited fields, and much more. This high performance sort/merge/record selection utility can be used as a stand-alone program or called as a subroutine from most languages.

Supports unlimited filesizes, multiple input files and fixed or variable length records. Many special file types are supported including Btrieve and dBASE. Up to nine sort control fields (ascending or descending), all common data types supported. Output files can be combinations of full records, keys or pointers, subsets of the input file fields, and literal values.

Written in assembly language for **high performance.** Example: 4,000 128 byte records sorted to give key and pointer in 30 seconds. **$149.**

On-Line Help™

A comprehensive utility for adding help windows to your programs. It provides efficient utilities and routines for interfacing your programs with the help system routines and help message libraries.

Help windows are displayed in a fraction of a second. You have total control over the contents of the window, its size and its position on the screen, including the display and border colors.

On-Line Help can be interfaced with interpreted Basic and all popular compilers. **$149.**

Scroll & Recall™

Allows you to conveniently scroll back through data that has gone off the top of your display screen. Up to 27 screens of data can be recalled or written to a disk file (great for documenting systems operations).

Allows you to easily recall and edit your previously entered DOS commands and data lines without re-typing.

Scroll & Recall is very easy to use. It's a resident utility that's always there when you need it! **$69.**

All programs IBM PC/XT/AT & MS-DOS compatible.

Visa, M/C, AMEX, Check, Money Order, COD or Purchase Orders accepted.

Quantity and Dealer Discounts Available

To order or to receive additional information write or call:

Opt-Tech Data Processing

P.O. Box 678 - Zephyr Cove, NV 89448

(702) 588-3737

Circle Number 16 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

YOUR COMPUTER LANGUAGE IS QUIETLY BREEDING REAL BATS IN YOUR BELFRY.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

WE'RE OUT TO SAVE ONE MILLION FRUSTRATED PROGRAMMERS

You're on a roll, really pumped, writing the best code you have ever written and then—AAARGHHH!

Freeze dried in your tracks because the language you're using just won't let you achieve what you can conceive.

And you wanted to be a programmer.

So your choices are:

1) write around the problem by creating six pages of emetic code...

2) leave out that incredible idea that really puts your stamp of excellence on this program or...

3) get yourself a world class headache (or a stroke) by dropping into assembler.

Whatever you choose, by now you feel the language is out to get you—because it is.

Sure, no language is perfect, but you have to wonder, "Am I getting all I deserve?"

And, like money, you'll never have enough.

Pretty dismal, huh?

We thought so, too.

So we did something about it.

We call it CLARION.

You'll call it incredible.

With CLARION you can write, compile, run and debug complex applications in a New York afternoon.

Even if you're in Savannah.

It gives you the power and speed to create screens, windows and reports of such richness and clarity you would never attempt them with any other language.

Because YOU would have to write the code.

With CLARION you simply design the screens using our SCREENER utility and then CLARION writes the source code AND compiles it for you.

In seconds.

Likewise, you can use REPORTER to create reports. Remember, only CLARION can recompile and display a screen or report layout for modification.

And with no time wasted.

All the power and facilities you need to write great programs, faster than you ever dreamed of.

Programs that are easy to use.

Programs that are a pleasure to write.

And to you that means true satisfaction.

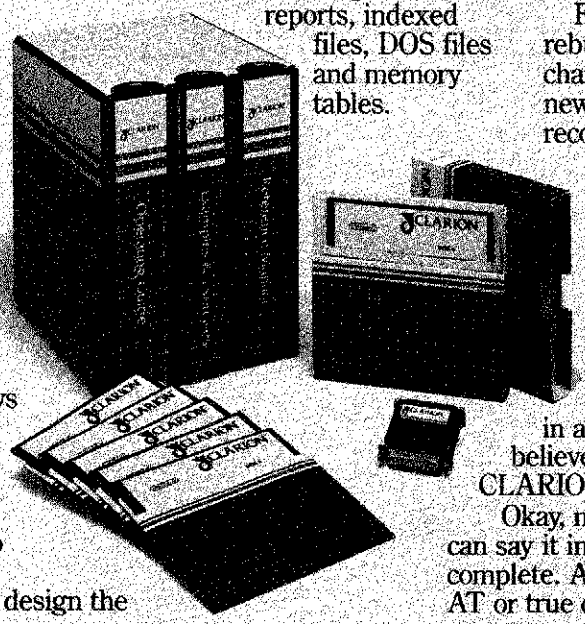
You've coveted those nifty pop-up help windows some major applications feature. But you can't afford the time and energy it takes to write them into your programs.

That's the way it used to be.

So we fixed that, too.

CLARION HELPER is an interactive utility that let's you design the most effective pop-up help screens that you can imagine. And they're "context sensitive," meaning you can have help for every field in your application.

Unlike the other micro languages, CLARION provides declarations, procedures and functions to process dates, strings, screens, reports, indexed files, DOS files and memory tables.



Imagine making source program changes with the CLARION EDITOR. A single keystroke terminates the EDITOR, loads the COMPILER, compiles the program, loads the PROCESSOR and executes the program. It's that easy!

Our data management capabilities are phenomenal. CLARION files permit any number of composite keys which are updated dynamically.

A file may have as many keys as it needs. Each key may be composed of any fields in any order. And key files are updated whenever the value of the key changes.

Like SCREENER and REPORTER, CLARION's FILER utility also has a piece of the CLARION COMPILER. To create a new file, you name the Source Module. Then you name the Statement Label of a file structure within it.

FILER will also automatically rebuild existing files to match a changed file structure. It creates a new record for every existing record, copying the existing fields and initializing new ones.

Sounds pretty complicated, huh?

Not with CLARION's documentation and on-line help screens. If you are currently competent in Basic, Pascal or "C" you can be writing CLARION applications in a day. In two days you won't believe the eloquence of your CLARION programs.

Okay, now for the best part of all. You can say it in CLARION for $295.00—complete. All you need is an IBM® PC, XT, AT or true compatible, with 320 KB of memory and a hard disk drive.

And we'll allow a full 30 day evaluation period. If you're not satisfied with CLARION, simply return it in its original condition for a full refund.

If you're not ready to take advantage of this no-risk opportunity, ask for our detailed 16 page color brochure. It vividly illustrates the elegance of CLARION. Consider it a preview of programming in the fast lane.

Either way, the call's a freebie.

SAY IT IN CLARION™

DEPT. A1PJ/1

1-800-354-5444

VISA MasterCard

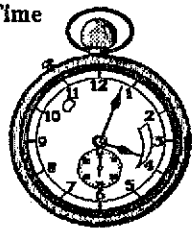
BARRINGTON SYSTEMS, INC. • P.O. BOX 5580 • POMPANO BEACH, FL 33074 • 305/785-4555

IBM® is a registered trademark of International Business Machines Corporation. CLARION™ is a trademark of Barrington Systems, Inc. ©1986 Barrington Systems.

Circle Number 55 on the Reader Service Card.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Along With Your Computer, Your Time is the Most Important Thing You Own. . .So Why Waste It?



Quilt Programmer Productivity Tools will help you manage your software projects and get control of your time!

The Software Revision Management System (SRMS)™ stores all versions of source code/documentation in a single ASCII file. Allows retrieval of any version of source and application of changes while recording when, why and where changes were made with no duplication of common code. Other features include: full DOS pathname, directory, and environment variable support to help manage library maintenance; complete typeset users manual with tutorial; extensive error recovery procedures, support for most programming languages and much more!
SRMS (Version 2.0) $125.00

QMAKE™ is not just another make utility for MS-DOS. Sure it will accept a control file specifying dependancies of your software system, and rebuild your system by compiling only those routines that need to be compiled. It will analyze all dependancies before rebuilding any part of your system, and do it in a way that is extremely fast. Support for macros, multiple entry points, command line parameters, and most languages is also included. QMAKE differs from other DOS make utilites in that it completely integrates with the Software Revision Management System to give you a powerful combination of productivity tools.

QMAKE (Version 1.0).....$ 99.00
SRMS + QMAKE Combination.....$199.00

QUILT COMPUTING
7048 Stratford Road
Woodbury, MN 55125
(612) 739-4650

MasterCard VISA

Circle Number 46 on the Reader Service Card

PL/D Compiler

PL/D (tm) sets new compiler performance standards for writing 8088/86/286 operating systems, utility programs, and stand alone applications. A 4.77 MHz 8088 with floppy disc compiled an 1800 line source file in 18 seconds. An 8 MHz HP Vectra 80286 with hard disk took 3 seconds! At run time, the INSEQU construct is used to define 1 of 2 byte machine language instructions that later may appear within expressions or as statements. 3 or more byte instructions use MACROs and appear as statements.

PL/D supports structured run-time facilities for FOR-NEXT, CASE, IF-THEN-ELSE-ENDIF, and WHILE, all of which may be nested. CASE allows an optional BUT clause which is taken if no CASE is selected. The ENDIF in the IF statement eliminates the "dangling ELSE" problem common to other languages. The { } pair serves as a generalized "address-of operator" that permits passing subroutine names as parameters, doing address computation on pointers, and many other functions, some with no equivalent in any other language.

PL/D has broad compile-time facilities. WHILE#, CASE#, FOR#-NEXT#, IF#-THEN#-ELSE#-ENDIF# and MACRO control structures direct what code is compiled during the actual compilation (not a pre-processor). ORG and COMN control placement of procedure and variables, respectively. MACRO writers have access to compiler indentation level, a macro stack, and the entire code workspace.

PL/D supports building modular programs through several levels of information hiding. The SYS statement reads in a separate source file. Each SUBR-ENDSUB statement pair defines a subroutine region where all names declared are local. The listing may be turned off and on globally or when specific names are used. Flipping back and forth between different processor listings is eliminated because the absolute code emitted, a called subroutine reference call, and proper indentation are integrated on one line.

Priced at $49.95, NOT COPY PROTECTED. MS-DOS 5 1/4" DS/DD format. 192 KB system.

DAIR COMPUTER SYSTEMS VISA MASTER CARD
3440 Kenneth Dr., Palo Alto, CA 94303
(415) 494-7081

Circle Number 5 on the Reader Service Card

as in sum(A, B, C). Of course, there is an infinite number of correct answers to these goals. In English, you are asking Prolog, "Give me two numbers that add up to some result." To avoid infinite looping, I used Turbo Prolog's random number generator to suggest at least one correct answer—the result that Cortesi said in his article would be "charming" if it were possible. In Turbo Prolog, it is.

Sum works by using the bound( ) and free( ) predicates to check whether a parameter has a value. This demonstrates a critical feature of Prolog—the ability to tell the difference between initialized and uninitialized variables, something you cannot do in Pascal except with pointers where the value Nil represents a non-existing address. Sum uses this idea to check for all possible sets of bound and free parameters, performing the necessary addition or subtraction for each condition.

Also in the SUM program are two predicates succ( ) and pred( ), which determine the successor and predecessor of a value. They are simple to program. Succ(X,Y) is true if sum(1,X,Y) is also true. (The :- stands for if.) Pred(X,Y) is of course true if Succ(Y,X) is true. Remember, all of these things are statements of fact—not directions for what you want Prolog to do.

These concepts are fairly easy to grasp, but to me the astounding thing is what happens when you type the goal Succ(X,25) and get X = 24. In other words, although you defined the successor function, Prolog finds the predecessor! And because both succ( ) and pred( ) are defined in terms of sum( ), you can type Succ(X,Y) and Prolog will suggest two adjacent ordered values—a completely unplanned result of Prolog's ability to fill in the blanks given a few facts. Such flexibility makes Prolog attractive, particularly in database programming.

Will Prolog Replace Pascal?

No. I don't think so. But it might replace BASIC. I haven't heard anyone else make that prediction, but if anything can move people away from BASIC it's Turbo Prolog. With its marvelous interface, a simple set of commands, an uncomplicated program structure, plus an interactive BASIC-like dialog, Turbo Prolog is just right for semi- and non-professional programmers.

But as good as it is, I cannot envision C, Pascal, and Modula-2 pros junking their old compilers and function libraries in favor of anyone's Prolog. My guess is you'll see many source code Prolog programs but not many compiled commercial ones. In fact, if the market isn't soon flooded with expert systems written in Turbo Prolog, I'll eat my disks. When true parallel processing computers arrive with a Prolog-type language and true parallel logic, then maybe Prolog will compete directly with Pascal and Modula-2 as a primary software development tool. I doubt that will happen soon, but I plan to stay current with Prolog just in case. Who knows what the gurus will do next? □

Tom Swan is the author of several books, including Pascal Programs for Data Base Management *and* Mastering Turbo Pascal *published by Hayden Book Company.* Mastering Turbo Pascal *is currently on B. Dalton's Top Ten bestsellers list. Address correspondence to Tom at*

P.O. Box 206
Lititz, PA 17543

Please enclose a self-addressed, stamped envelope if you want a personal reply.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Brand New From Peter Norton

A PROGRAMMER'S EDITOR

that's *lightning fast* with the *hot* features programmers need

A WONDERFUL EDITOR FOR PROGRAMMERS FROM PETER NORTON

THE NORTON EDITOR™

only **$50**

***E**asily customized and* saved ▲ *Split-screen editing* ▲ *A wonderful condensed/outline display* ▲ *Structured programming features:* ● *Auto-indenting for* Pascal *and* C ● *A* hot *"find matching punctuation" feature* ▲ *Great word features too:* ● *Paragraph reformat* ● *Word wrap, word action* ● *You can write your programs* and *your manuals with this one editor* ▲ *Mouse support, too* ▲ *Great for* assembler, *fabulous for* Pascal *and* C

A STEAL AT $50!

Direct from the man who gave you *The Norton Utilities, Inside the IBM PC,* and the *Peter Norton Programmer's Guide.*

"This is the programmer's editor that I wished I'd had when I wrote my *Norton Utilities.* Now I've got it for you to use. You can ***program your way to glory*** with *The Norton Editor.*"

Peter Norton

Peter Norton Computing, Inc., 2210 Wilshire Boulevard, Santa Monica, CA 90403, 213-453-2361. Visa, Mastercard and phone orders welcome.

The Norton Editor™ is a trademark of Peter Norton Computing, Inc. © 1988 Peter Norton Computing

Circle Number 17 on the Reader Service Card

New Version 2.0

Complete C Programs in Half the Time, with *Instant-C*™

You can create programs much faster with Instant-C than with conventional programming tools. How? Because Instant-C is a high-performance interpreter, there are **no compile or link delays.** Change your program, then test it immediately. No matter how large your program, the turnaround time is just seconds.

"Instant-C means instant gratification."—*PC Magazine,* **Editor's Choice** for best C interpreter.

Powerful **source-level debugging** saves your time. Conditional breakpoints, single-stepping by statement, source code backtraces, data monitoring, and many other debugging features make it easy to wipe out bugs quickly. Direct execution of any statement or function makes testing a breeze.

"The resulting debugging and testing capabilities are fantastic and the detailed trace/debug/display commands make it easy."—*The C Journal*

Instant-C checks pointer references for reasonableness, and checks that array indexes are within declared bounds. This **run-time checking** stops your program as soon as errors occur, for easiest debugging.

Not only does Instant-C help you quickly change, test, check and debug your code, but it runs your program **fast enough for real-time** applications.

"It is much faster than any of the other products mentioned and was the only one able to complete the standard SIEVE in a reasonable time. Clearly, this high speed allows much more complex problems to be attacked with Instant-C than with any of the other products discussed."—*Computer Language*

Immediate feedback and precise diagnostics make Instant-C great for learning C. Full K&R and the ability to **link compiled object code and libraries** (Lattice and Microsoft) makes Instant-C compatible with your existing programs.

Instant-C makes all parts of the programming task as fast as possible.

"Clearly, Instant-C is the performance champion."—*PC Tech Journal*

Version 2 works with MS-DOS and PC-DOS, and has a full 31 day **money back guarantee.** Instant-C is only $495. Order today! Call or write for full information.

Rational Systems, Inc.
P.O. Box 480
Natick, MA 01760
(617) 653-6194

Circle Number 20 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

C DYNAMO

FREE! PCUNIX™

(WITH ANY PURCHASE OF $250 OR MORE. LIMITED TIME OFFER)

LIBRARY: *325 fully tested functions screen handling/graphic, cursor/ keyboard/data entry, string handling, status and control, utility/DOS/BIOS/ time/date, printer control, 4 disks* $129.95

WINDOWS: *The most powerful windowing package available, overlays, borders, pop-up menus/ help windows, zap instantly on/off screen, status lines, horizontal/ vertical scrolling, color control/ highlighting, word-wrap, files to windows, keyboard to windows 3 disks* $129.95

SUPERFONTS FOR C: *Dramatic, high impact screens. Use our character & image libraries and functions, or create your own. Font & Function Library* $49.95

B-TREE LIBRARY: *Fast indexing B-tree. 16 million each: keys and records, unlimited keys per file, variable length records. PC-DOS, UNIX, XENIX, CP/M.* $79.95

ISAM DRIVER: $49.95

COMPATIBILITY: *PC/XT/AT; Full K&R, CI-C86/De Smet/ Lattice/ Microsoft 3.0/Aztec/others.*

VALUE: *All source code, No royalties. Best documentation available.*

No matter what else you have, you need these.

C-TERP: *Simply the best C interpreter available. See C Journal Summer 1985*
C-TERP (specify compiler) $299.95

PCUNIX: *Multi-tasking, networking multi-user. With source code* $99.95
™Wendin Software

COMBINE AND SAVE!

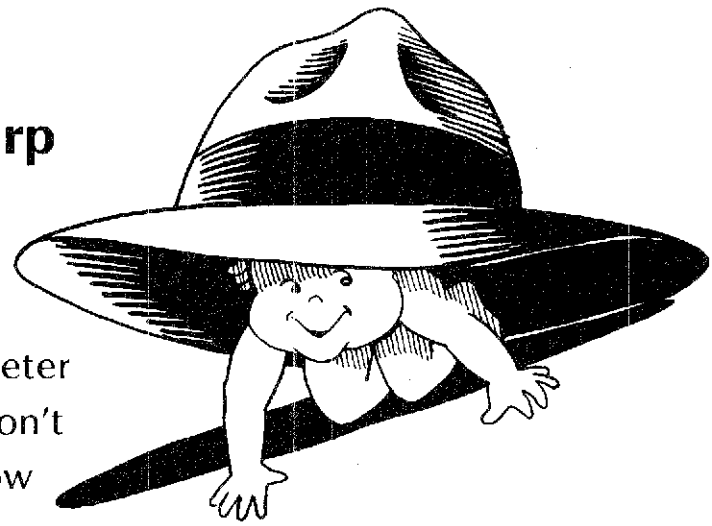
C LIBRARY plus C WINDOWS
BOTH for only $179.95
+ SUPERFONTS FOR C . $199.95
+ B-TREE and ISAM $299.95
(A $440 VALUE)
C LIBRARY plus C WINDOWS
+ SUPERFONTS
+ C-TERP (a $610 value) $459.95
All (a $740 value) $549.95

Entelekon

12118 Kimberley, Houston, TX 77024
713-468-4412
VISA MASTERCARD CHECK COD

Circle Number 9 on the Reader Service Card

C-terp



The C Interpreter You Won't Outgrow

C-terp will grow with you as you progress from novice through professional to guru. Unbelievable, but true, the easiest-to-use C interpreter will provide you with the most advanced programming features for upward growth. Our exclusive **object module support** enables you to add libraries (like HALO, PANEL, Windows for C, etc., or your own homebrew libraries) to C-terp as you add them to your computing repertoire. Use C-terp as a microscope on your libraries! Flip a bit and allow our **software paging** (NEW) to handle those big jobs! There are no fixed-size tables to overflow, and C-terp can be configured for different screens and screen adapters (NEW). With multiple modules and **full K&R support,** we offer a dream C environment.

- Our new improved **configurable editor** competes with anything going.
- Speed -- Linking and semi-compilation are breathtakingly fast.
- Convenience -- Errors direct you back to the editor with the cursor set to the trouble spot.
- Symbolic Debugging -- Set breakpoints, single-step, and directly execute C expressions.
- Compatibility guaranteed -- batch file to link in your compiler's entire library. Supported compilers include: **Computer Innovations C86, Lattice C, Microsoft C 3.0, Mark Williams C86,** and **Aztec C.**
- Many more features including batch mode and 8087 support.

What Our Users/ Reviewers Are Saying

"... easy to use, powerful, and a timesaver."
"... we absolutely LOVE C-terp."
"... has restored my faith in interpreters."
"... a programmer's dream."
"... wonderful technical assistance."
"... increased our productivity by a factor of 40."
"... the best C product ever, in any category."

- **Price: $300.00 (Demo $45.00) MC, VISA**

Prices include documentation and shipping within U.S. PA residents add 6% sales tax. Specify compiler.

- C-terp runs on the IBM PC (or any BIOS compatible machine) under DOS 2.x and up with a suggested minimum of 256 Kb of memory. It can use all the memory available.

* C-terp is a trademark of Gimpel Software.

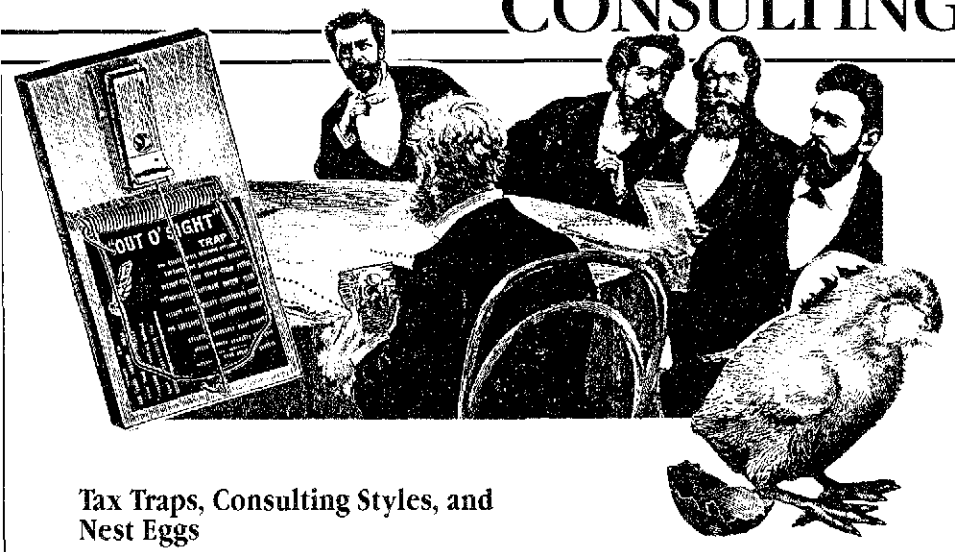
GIMPEL SOFTWARE

3207 Hogarth Lane • Collegeville, PA 19426
(215) 584-4261

Circle Number 11 on the Reader Service Card

Brettie Mathews, a techie ahead of her time, dreams of the proportional control concept.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED



Tax Traps, Consulting Styles, and Nest Eggs

by Paul Barkley

California Brouhaha

I don't usually pay much attention to what any legislature is doing, much less one in another state, but there's an issue brewing in California that many people claim will have a dramatic—and undesirable—impact on consultants. It seems that the state has taken a dim view of employees masquerading as independent contractors, and while one branch is trying to pass some legislation, another branch is busy levying large fines on companies who have hired the questionable people.

The Independent Computer Consultants Association is very interested in this activity because many members are contract programmers and are heavily impacted by this. The legislation is really aimed at contract programmers who are not readily distinguishable from employees. As I've said in earlier columns, if contract programmers want to call themselves consultants, that's okay with me (and I welcome their readership and comments), but there are some fundamental differences in the client relationships of each group.

Contract programmers are generally characterized by working at the employer's site, on the employer's equipment, for extended periods of time, often 6 months to a year. They follow the same rules as regular employees and work the same hours. In fact, the only difference between many contract programmers and employees is that the contract programmer's termination has supposedly been built in to the contract, and because of his or her special status the contract programmer probably gets paid more "salary," although with no benefits.

In extreme cases, the contract is renewed repeatedly, and the contract programmer gets holidays and vacations like the rest of the staff. That sure looks like an employee to me, and that's what it looks like to the state of California, too.

The criteria that they're proposing seem pretty innocuous to me (easy to say when it's not my job on the line, huh?). They want an independent contractor to perform services through a separately established business; have a real contract and perform to the terms of the contract; maintain a separate business location; utilize skills different from the client's employees; perform services related to a specified result within a specified time; receive no vacation pay or health or retirement benefits; and pay federal self-employment tax.

Of these criteria (more specific than I've listed), only *five* must be met. This takes care of cases, for example, where a consultant is hired to perform some ill-defined advisory services, or where a consultant performs services that aren't especially different from employees', but otherwise meets the criteria.

The issues are a little more complex than this, of course, but the state is trying to drum up some more revenue by collecting the employer's share of payroll taxes that aren't collected when an employee is disguised as an independent contractor. The IRS has looked unfavorably at this sort of thing for years—the employers of programmers are not the first people with the bright idea of saving some money by redefining the status of an employee.

My advice? If you really are an independent consultant, you won't have much to worry about.

On a related note, the computer consulting industry is talking more about certification. The argument is that we need to regulate ourselves before the government steps in and does it for us. Having taken a look at the present certification procedures, I can't say I see much hope that they'll create a fair measure of computer competency. The last time I looked, a working knowledge of COBOL was a prerequisite. I suspect that everyone will feel that his or her special knowledge can't be measured by someone else's standards. I know that I wouldn't want the job of setting up the standards, either.

One very nice thing about standards—for those on the inside—is that they raise a barrier to entry in a profession, and this can be expected to drive prices up. Once you're in, you've got it made. Perhaps it's time we established a standard for magazine writing. To become certified, you'd have to prove a knowledge of point sizes, typefaces, grammar, and libel laws. I know a lot of people already who are often referred to as certified, but that refers to their mental state and not to their profession. What do you think about certification?

A Different Consulting Style

Rex Jaeschke, who writes the C column for *PJ*, gave a presentation recently to a local consultants' group and had some very interesting points to make. Several of them are in sharp contrast to my own practice and illustrate the fact that there are many ways to achieve a successful consulting business.

Rex is adamant about working from an office in his home, never employing anyone, and never wearing a tie. I prefer a regular office, think employees are the greatest thing since draft beer, and always wear a tie. (Of course Yogi Bear always wore a tie, too, but you get the idea—business suits.)

My attitude about clothes is that I can't do any better work in a suit than I can in jeans (and maybe worse!), but if someone is going to pay me a lot of money per hour, the way they pay an attorney, I ought to dress like an attorney. (I said dress like an attorney, not act like an attorney.) Rex's attitude is that he's a high-priced technician called in to solve a problem, and therefore, it's perfectly proper that he look like a technician.

I've had a lot of experiences that have convinced me that you generally get a lot farther when you dress for the occasion.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

When you dress poorly, insecure people will give you a hard time. Those same insecure people are intimidated by well-dressed people and not only won't impede your progress, but will often go out of their way to be helpful. (Disgusting, isn't it?)

I also think dressing appropriately is a civilized sign of respect for those you're dealing with and that clients generally appreciate it. Since wearing a vest and coat in the middle of summer is so clearly asinine, as are high heels for women, our culture has decided that these constitute very polite behavior.

The only time I've seen this backfire is when a friend of mine showed up for a casual party one summer 15 years ago, wearing a dark blue suit and white shoes. All the bell-bottomed hippies there marveled at the white shoes, and they still bring it up as the only thing they remember about that person. Better not wear a tux to visit your clients.

Think about the image that you present. Some professionals, such as interior designers, rock musicians, or sewage tank inspectors, can get away with unusual dress. Can you?

Rainy Day People

I come into contact with a lot of people planning to start consulting businesses, and of course each person has a number of concerns.

Your biggest concern should be whether you're going to be able to eat. Most people contemplating going out on their own already have the necessary technical skills, or are in the process of acquiring what they need. Most people worry about marketing because they know that it's a subject far removed from the technical aspects of computers, but even marketing skills can be learned. What you *really* need is a fat contract.

What makes the big difference between successful and unsuccessful consulting, in my opinion, is acquiring sufficient continuing business. The most cheerful, confident consultants usually have some long term relationship established with a large firm to provide services that are not so different from what they may have provided as an employee.

The ideal contract seems to be for about half time work, either a 20 hour week or a certain number of weeks per month or quarter. Since the billing rate should be *at least* twice as high as the salaried rate, half time gives the consultant the same income with a lot of spare time to get into other projects. Sounds good, doesn't it?

Many consultants got into the business because they wanted more spare time (in addition to getting out from under the corporate thumb), so they don't look very hard to fill that slack time. They're comfortable.

Other consultants, though, will recognize the need to build up a reserve fund to carry them through any lean periods, and they'll work as hard as they can while the jobs are still there. They'll maintain their marketing efforts even when they're working 60 hours per week because they know that it's too late to start marketing once you've run out of work.

The key to all this, however, is financial stability. You *must* have a solid contract or a lot of operating cash before you begin. Plan on setting aside at least six months' worth of expenses (personal and business) and try from day one to keep that reserve. When it diminishes, try even harder to build it back up. If it gets down to a month or so, you'd better consider going back to work for someone—you've about had it at that point.

object $49 — BASIC WINDOWS — asm source $149 *

Create "instant" pull down menu's or pop-up help windows using any color combo's desired. Basic Windows uses a smart window buffer manager which does not use available BASIC memory! Also allows async operations for a dozen simultaneous windows. Screen save/restore, window create/restore/toss, window print using word wrap & other controls, Input string$ from window, clear/scrollup/ scrolldn/clearEOL&EOS. Includes 2.5k code, doc, test program. For IBM & Compaq BASICA or IBM & MSQUICK compilers on Dos 2 or 3.

object $59 — C DATA ENCRYPTION — c source $249 *

A complete security system using DES (Data Encryption Standard) per U.S. government standard FIPS PUB46 written in Microsoft C. The 3 modes are supported: Electronic code book, cipher feedback, and cipher block chain. Also included is data compression and telecommunications formatting for transmission and storage via any network medium. Send your electronic mail encrypted via Telemail, MCI mail, Easylink Telex, or BBS ?!!

object $99 — COMMX — consolx $69 option

COMMunications eXchange provides multi-terminal emulation, multi-protocol file transfers, unattended control macros and multi-site dial & parameter directory. Emulates VT100/102/52, IBM3101-1x, WYSE50, TV950, HP2624/45, ADM3/11, ADDS R60 (all character modes). Terminal modes support full color controls with optional 25th status line showing time and caps/num lock status. File XFER for ASCII, KERMIT, XMODEM, and COMMX (faster than KERMIT using COMMX/Mainframe). CONSOLX option provides remote PC operation.

HAWKEYE GRAFIX Inc
3415 Hyde Park Drive
Clearwater, FL 33519
call => 813-786-8161

* discounted thru Sept 1986

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Have I taken my own advice? No. Did I start with a lot of cash or a solid contract? No. Can you survive this way? Yes. Is it any fun? Absolutely not—it produces unbelievable anxiety and depression. It destroys marriages and alienates other family. The only positive benefit of starting with insufficient capital and no major contract is that you develop ferocious accounts receivable policies that will serve you well in better times—if you survive. If you're not certain that you're *really* an entrepreneur under that corporate facade and are prepared to pay the price, don't do it. I've seen a number of people who wanted the freedom but who couldn't pay the price go back to their previous positions. There's nothing wrong with that, of course, but get your finances and contracts lined up *before* you make the plunge. If you want to weather the rainy seasons, pile up the green sou'westers ahead of time. Otherwise, you'll get wet and hungry.

Drop me a line with your thoughts, comments, or questions about consulting. □

Paul Barkley
The Barbican Company
6845 Elm Street, Penthouse One
McLean, VA 22101

CompuServe 74776,25267

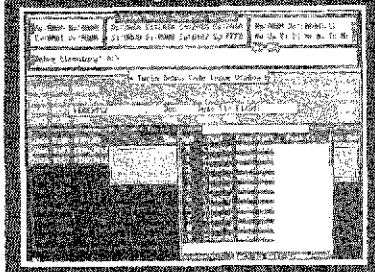
SPEEDWARE™
A DIVISION OF AMERICAN AFFORDABLE SOFTWARE CORPORATION

PRESENTS

Turbo Debugger™ Features
- User-definable custom windows — Change window sizes or colors
- Single-step tracing with "Back-Tracing" up to 100 instructions
- In-Line relocating Assembler with non-destructive insert capability
- Fully Symbolic, Create Comments and Line Labels, Keyboard macros
- Dual Monitor support, with Screensave
- Specify several Breakpoint conditions with the built-in Assembler
- Use NMI Breakout switch or User-definable hotkeys
- Read, Write and Modify .EXE files

Turbo Editasm™ Features
- Integrated WordStar compatible Editor
- Integrated MicroSoft compatible Macro Assembler
- Full 8088, 8086, 8087, 80186, 80287, 80286(R) & (P) support
- Full Macro expansion capability
- Built-in Calculator & Xreference Capability
- Limited DOS command support
- Supports 16 Memory Assembly

To maximize your programming potential order the ultimate "Turbo Team" today — the first well-tuned, high-speed assembler/debugger for the IBM P.C. and compatibles family.



$99.00 Turbo Editasm™
$89.00 Turbo Debugger™

THE TURBO TEAM™
Inside California (800) 334-3664 • Outside California (800) 443-6190

$99.00 ☐ Turbo Editasm™
$89.00 ☐ Turbo Debugger™

Name _____
Shipping Address _____
Amount Enclosed _____
Payment VISA MC BANK DRAFT CHECK
Credit Card Exp Date _____
City _____
Card # _____
State _____ Zip _____
Telephone _____

Dealer and Education Discounts Available.

Speedware™
9719 Lincoln Village Dr., Ste. 205
Sacramento, CA 95827

TURBO PROGRAMMERS—

TDebug Plus™...

...CUTS DEBUGGING FRUSTRATION.

TDebugPLUS is a **new**, interactive symbolic debugger that integrates with Turbo Pascal to let you:
- ***Examine and change variables*** at runtime using symbolic names – including records, pointers, arrays, and local variables;
- ***Trace and set breakpoints*** using procedure names or source statements;
- ***View source code*** while debugging;
- ***Use Turbo Pascal editor and DOS DEBUG commands.***

TDebugPLUS also includes a special MAP file generation mode fully compatible with external debuggers such as Periscope, Atron, Symdeb, and others – even on programs written with Turbo EXTENDER.

An expanded, supported version of the acclaimed public domain program TDEBUG, the TDebugPLUS package includes one DSDD disk, complete source code, a reference card, and an 80-page printed manual. 256K of memory required. Simplify debugging! $60 COMPLETE.

TURBO EXTENDER™

Turbo EXTENDER provides you the following powerful tools to break the 64K barrier:
- ***Large Code Model*** allows programs to use all 640K without overlays or chaining, while allowing you to convert existing programs with minimal effort; makes EXE files;
- ***Make Facility*** offers separate compilation eliminating the need for you to recompile unchanged modules;
- ***Large Data Arrays*** automatically manages data arrays up to 30 megabytes as well as any arrays in expanded memory (EMS);
- ***Additional Turbo EXTENDER tools*** include Overlay Analyst, Disk Cache, Pascal Encryptor, Shell File Generator, and File Browser.

The Turbo EXTENDER package includes two DSDD disks, complete source code, and a 150-page printed manual. Order now! $85 COMPLETE.

TURBOPOWER UTILITIES™

"If you own Turbo Pascal, you should own TurboPower Programmers Utilities, that's all there is to it." Bruce Webster, ***BYTE Magazine***

TurboPower Utilities offers nine powerful programs: Program Structure Analyzer, Execution Timer, Execution Profiler, Pretty Printer, Command Repeater, Pattern Replacer, Difference Finder, File Finder, and Super Directory.

The TurboPower Utilities package includes three DSDD disks, reference card, and manual. $95 with source code; $55 executable only.

ORDER DIRECT TODAY!
- ***MC/VISA Call Toll Free*** 7 days a week.
  800-538-8157 x830 (US)
  800-672-3470 x830 (CA)
- ***Limited Time Offer!*** Buy two or more TurboPower products and save 15%!
- ***Satisfaction Guaranteed*** or your money back within 30 days.

For Brochures, Dealer or other Information, PO, COD – call or write:

TURBO Power

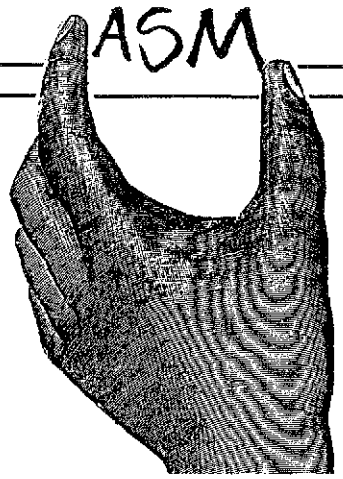
478 W. Hamilton #196
Campbell, CA 95008
(408) 378-3672
M-F 9AM-5PM PST

The above TurboPower products require Turbo Pascal 3.0 (standard, 8087, or BCD) and PC-DOS 2.X or 3.X, and run on the IBM PC/XT/AT and compatibles.

Circle Number 72 on the Reader Service Card

Circle Number 25 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED



Search and Display

by Dan Daetwyler

I'm tempted to name this article YASU (yet another search utility). I've encountered quite a few utilities that will permit searching an ASCII file for a specific text string, but none would do quite what I needed.

This program performs the traditional "search" for the specified text string of an ASCII file, but with a slight twist. This program will search *all* of the files matching a "wild card" filename on the default directory, for as many as 10 different text strings.

The Environment

Before going further with the program description, I'd better describe the environment it's set to "fit." I tend to develop relatively large, complex application systems, and since I'm an Assembler bigot, and believe in "modularity," I usually end up with something between 100 and 200 different source modules in a system. My definition of "modularity" is primitive: If the assembly listing takes more than two pages to print, the module is too big! Typically, each module is a single procedure although I will violate that concept when I have two or three very short procedures that are logically related.

I might suggest that this concept is equally applicable for "high-level language" programmers. We have reasonably good and powerful "linkers" now, so let the linker work and save yourself a fair amount of compile and listing time.

Search

So what has this to do with "search"? Well, consider my problem. I may have 150 source modules, and scattered throughout them are many references to a data area named ZONE. Because of a change, I need to change the length of ZONE from a byte to a word. Or, more commonly, I find a "bug" in a module named SHIP. What procedures call this module? All must be reviewed for impact if any changes are made in SHIP.

With the utility shown here, all I have to do is execute the utility and respond with ZONE (as a search mask in the first case), or CALL SHIP (in the second case), and then give the file mask as "*.ASM." The utility will scan the entire source directory, searching every assembly source module for references to the search masks. When it finds a match, it will display (or print) the module (file) name in which it found the match and the first 65 characters of the source line in which it was found. In fact, I could combine the two searches into one since the utility permits input of up to 10 search masks per execution. By giving the full filename instead of a "wild card" filename, I may restrict the search to a single file although this is rarely useful. The utility is fast enough that it will search and display its matches much faster than you can read them, so it will pause at the "end" of each screen (in display mode), to allow you to read the results. Pressing any key will cause it to continue.

"Glitches"

Two minor "glitches" exist in the program. First, the search masks are accepted as exact character string matches. If you specify "AX" as a search mask, you'll match on lines that include "MAX" as well as "AX." If you want to match on " AX " (leading and trailing blanks), input the mask with leading and trailing blanks. Many editors substitute "tab" characters for blank strings, so you must know your editor to be sure you're getting the results you expect from this utility. For example, the source statement "CALL SHIP" commonly would have a "tab" instead of a blank between CALL and SHIP. If your editor works this way, enter CALL tab SHIP, rather than CALL blank SHIP, or you won't get the results you expect.

A second "nit" may be encountered if you are searching large ASCII files. The utility, for simplicity, will read the first 40,000 bytes of a file and then continue to process the file in 40,000 byte "chunks" until it encounters an end of file. If you're unlucky enough to split a search string between two of these "chunks," it will not be "matched" by the utility. This bug could easily be removed, but most source files are considerably smaller than 40,000 bytes (I'd hate to wait for the thing to compile if it's larger than this), so the "nit" is unlikely to be encountered in most development environments.

The Source

The accompanying source file for this utility is simple and adequately commented. It makes use of several DOS calls, but these are the commonly used DOS services: string input and output, character input, and search directory for first/next matching filename. The main procedure SRCHDSP does the initialization and "housekeeping," calling subordinate procedures for much of the processing. DOMASK is the primary processing procedure, reading a file and then searching for matches on the input masks. It uses subordinate procedure MATCH for checking each source line against the search masks and subordinate procedure SKPLNE to skip the remainder of the current line when a match has been found.

MATCH checks each input mask against each character of a source line by use of the procedure STRING. STRING, in turn, does an "early out" test of the current source character against the first character of a search mask. If that comparison fails, it steps to the next search mask and tries again. When all search masks have been checked, STRING makes a "fail" exit if no matches have been found. If the first characters match, STRING proceeds with the comparison of the entire mask string. If this comparison results in an "equals" condition, STRING calls PRINT to cause the source line to be displayed and makes an immediate "match" exit.

PRINT simply outputs the source filename, followed by the first 65 characters of the source line. Several tests are made within PRINT, based on the output media requested. Keyed off the user's request for "P" (print) or "D" (display) output, the procedure sets the proper DOS call register and controls the "page" length.

The two remaining procedures (GETMSK and SKPLNE) are used to "stack" the input search masks specified by the user and to skip the balance of the current source line without further checking for matches once the source line has been output.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

No "external" routines are used, so link of the object module is absolutely "plain vanilla."

And for those of you who are observant, I do "practice what I preach." This is a considerably larger than normal source module (for me) simply because magazine editors worry about column inches and breaking this thing into my more normal four source modules would require several additional column inches, which wouldn't add appreciably to your use or understanding of the module.

Improvements

Several "improvements" are possible. A more sophisticated mask input and search process might relieve some of the ambiguities that can be encountered with the search masks. The utility is now "case sensitive," and you might prefer to eliminate case considerations for matching. You probably won't be able to "improve" on its speed (to any appreciable degree, at least) because it's fairly fast right now. That isn't a challenge, by the way, but is intended to help you put your effort where it will achieve significant results. Feel free to extend, or modify, the utility in any way you like. With this publication, I have placed the source in "public domain," with no restrictions whatever. Have fun!! □

Dan Daetwyler
Route 5, Box 518A
Springdale, Arkansas (AR) 72764
(501) 756-0212

We received the following biographical note from Dan:

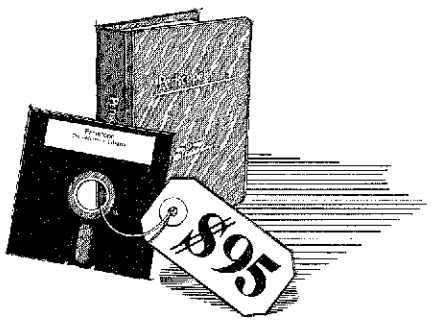
Dan Daetwyler (D-Squared) graduated from Kansas State College of Pittsburg in 1959 with a B.S. in Physics and then accidentally fell into programming. Aimed by interest and training towards the Electronics Engineering field, he was something less than pleased to find that his first assignment was in "mushware." Twenty plus years later, at retirement, he discovered that the disease still had not been cured, so he continues to "scratch the itch" by developing "custom systems" for small business and writing useful (he hopes) tools for programmers. He is the author of COVER, FSED, SDA, and EILY, all "user supported freeware" that have been fairly widely distributed. Prior to retirement, he was employed by "the Itty Bitty Machine Company" (better known as "Big Blue"), holding a variety of positions, at least one of which was appropriately titled: "Non-functional Manager." He currently lives in an "earth sheltered" home (hibernation cave) on a bluff overlooking a large lake in northwest Arkansas and is frequently heard to comment that the view is considerably more pleasant than George Washington Parkway during rush hour. He shares his cave with his wife, two daughters, and Buddy the Beagle and is convinced he can demonstrate absolute proof of the accuracy of the old story that the Chinese ideogram for trouble is a line drawing of two women under one roof. He's convinced that this is also an example of a "power law" relationship.

Code follows.

```
 PAGE 81,132
 TITLE SRCHDSP - Library Search Utility
 SUBTTL V2.0 - Dan Daetwyler - April 1986
;
DATA    SEGMENT PARA PUBLIC 'DATA'
;
MMSG    DB      13,10
        DB      'Enter search masks (Null to stop)'
CRLF    DB      13,10,'$'
FMSG    DB      13,10,10,10
        DB      'Enter file mask:  $'
OMSG    DB      13,10,10,'Printer or display (P/D): $'
WMSG    DB      'Press any key to continue$'
;
HAND    DW      0                       ;Source file handle
MCNT    DW      0                       ;Mask count
SKIP    DB      0                       ;Skip to EOL flag
IBUF    DB      20                      ;DOS input buffer (masks)
ILEN    DB      0
IDAT    DB      21 DUP (?)
SFINP   DB      40                      ;DOS input buffer (file)
SFLEN   DB      0
SFIL    DB      41 DUP (?)
FBUF    DB      21 DUP (?)              ;"Find" DTA area
FATR    DB      ?                       ;File attribute
FTIM    DW      ?                       ;File time
FDAT    DW      ?                       ;File date
FSIZ    DW      ?                       ;File size
        DW      ?
FNAM    DB      13 DUP (?)              ;File name
LCNT    DB      0                       ;Output line counter
LMAX    DB      ?                       ;Max lines per "page"
OMODE   DB      ?                       ;DOS call type for output media
MODE    DB      ?                       ;P/D flag for output media
CNTX    DB      0                       ;Continuation switch
;
XMASK   DB      10 * 21 DUP (?)         ;Mask storage
        DB      10                      ;Line feed to terminate scan
BUF     DB      40000 DUP (?)           ;Input file buffer
;
DATA    ENDS
;
STACK   SEGMENT PARA STACK 'STACK'
        DB      20 DUP ('STACK')
STACK   ENDS
;
CODE    SEGMENT PARA PUBLIC 'CODE'
        ASSUME  CS:CODE,DS:DATA,ES:DATA
;
SRCHDSP PROC    FAR
        PUSH    DS
        XOR     AX,AX
        PUSH    AX
        MOV     AX,DATA
        MOV     DS,AX
        MOV     ES,AX
        MOV     DI,OFFSET XMASK         ;Point to mask save area
        MOV     MCNT,0                  ;Clear mask count
        MOV     CX,10                   ;Set for a max of 10 masks
SMLP:   MOV     DX,OFFSET MMSG          ;Prompt for user search mask
        MOV     AH,9
        INT     21H                     ;DOS line output
        CALL    GETMSK                  ;Get user mask
        JC      MDNE                    ;User indicated "end of masks"
        ADD     DI,21                   ;Step to next save area
        LOOP    SMLP                    ;Loop through prompt for mask
MDNE:   MOV     DX,OFFSET FMSG          ;Prompt for file mask
        MOV     AH,9
        INT     21H                     ;DOS line output
        MOV     DX,OFFSET SFINP         ;DOS line input
        MOV     AH,10
        INT     21H
```


PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Periscope Is On Sale!



"A marvelous debugging tool..." says Peter Norton.

SUMMER SALE. Through August 31, the new software-only Periscope II-X is yours for just $95! Periscope II, normally $145, is only $125!

NEW MODEL. Periscope I includes both a break-out switch for "spontaneous" debugging and a RAM board to protect the debugger from programs that overwrite memory. Periscope II includes a break-out switch that doesn't require an extra slot to install. *And the new Periscope II-X is Periscope II without the break-out switch. So you can now get Periscope with no hardware at all!*

NEW RELEASE. The new version 2.1 enhances all models with many of the features users have requested. Call for details.

CHECK OUT THE REVIEWS.

"Periscope strikes a balance of power and features that will please serious PC programmers."
—Jeff Duntemann, PRODUCT OF THE MONTH, *PC Tech Journal*, 1/86

"Periscope's diverse features, affordable price, and portability place it in a class by itself." —Ward Christensen, "Breaking out with Periscope," *PC Tech Journal*, 3/86

"Periscope represents the finest software debugger available in its class." —Andrew Fried, *Computer Shopper*, 4/86

REQUIREMENTS. An IBM PC, XT, AT or close compatible; 128K RAM; DOS 2.0 or later; one disk drive; and an 80-column monitor.

Periscope II-X, Software Only ..... $ 95*
Periscope II, with Switch $125*
Periscope I, with Board & Switch .. $295
*Prices good through August 31, 1986

Take advantage of the low sale prices while they last . . . Order your Periscope today! Call toll-free:

800/722-7006

30-Day Money-Back Guarantee

Shipping-$3 UPS ground; $5 COD; $8 Canada; $24 Foreign. We accept qualified PO's in the U.S.

Get Your Programs Up and Running;

UP PERISCOPE!

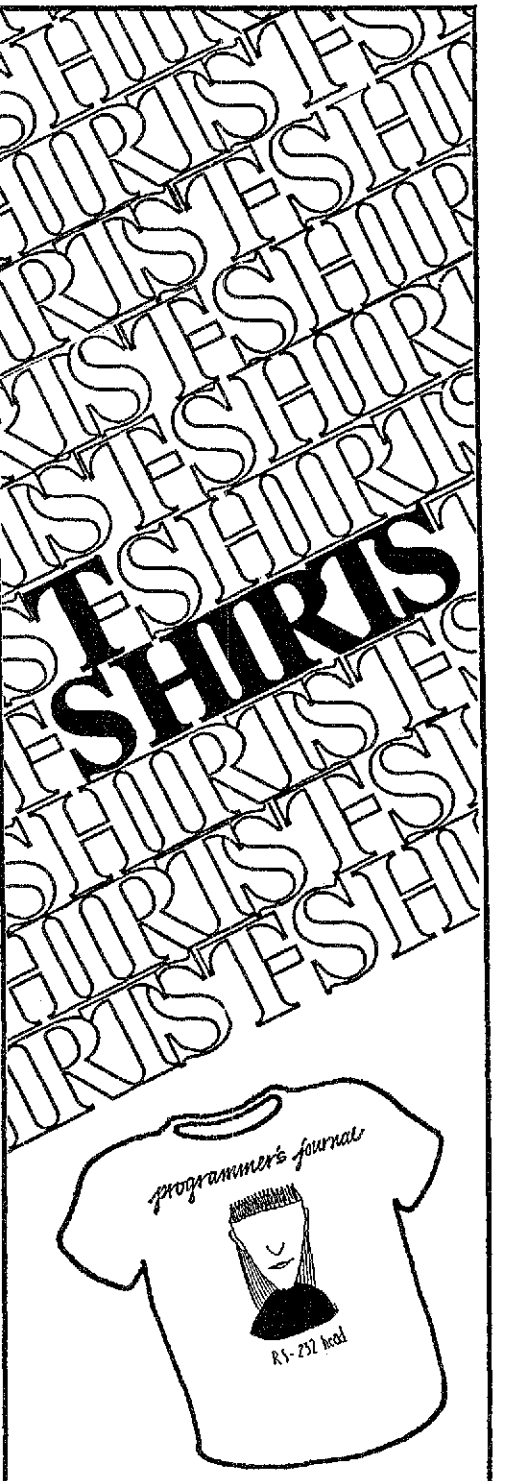
Data Base Decisions • (404) 256-3860
14 Bonnie Lane • Atlanta, GA 30328

Circle Number 6 on the Reader Service Card.


```
        MOV     DI,OFFSET SFIL          ;Point to file mask
        MOV     AL,SFLEN                ;Step by length
        XOR     AH,AH
        ADD     DI,AX
        XOR     AL,AL
        STOSB                           ;Force "null" terminator to name
OLP:    MOV     DX,OFFSET OMSG
        MOV     AH,9
        INT     21H                     ;Prompt for output media
        MOV     AH,1
        INT     21H                     ;Wait for user input
        AND     AL,5FH                  ;Force upper case
        CMP     AL,'P'
        JE      MDOK                    ;Printer mode
        CMP     AL,'D'
        JNE     OLP                     ;Invalid mode
MDOK:   MOV     MODE,AL                 ;Save media requested
        MOV     DX,OFFSET CRLF
        MOV     AH,9
        INT     21H                     ;Home cursor
        CMP     MODE,'P'                ;Check for printer mode
        JE      PRTR
        MOV     OMODE,2                 ;Set for DOS console output
        MOV     LMAX,23                 ;  and max lines
        JMP     SHORT GOON
PRTR:   MOV     OMODE,5                 ;Set for DOS printer output
        MOV     LMAX,58                 ;  and max lines
GOON:   MOV     AH,1AH
        MOV     DX,OFFSET FBUF
        INT     21H                     ;DOS Set DTA
        MOV     AH,4EH                  ;DOS Find "first"
        JMP     DOSRCH
BIGLP:  MOV     AH,4FH                  ;DOS Find "next"
DOSRCH: MOV     DX,OFFSET SFIL          ;Points to file search DTA
        XOR     CX,CX                   ;Normal mode
        INT     21H                     ;DOS Search for file
        JNC     FOK                     ;File matched file mask
FAIL:   CMP     MODE,'P'                ;See if printer output
        JNE     DOSRET                  ;No, so just exit
        MOV     AH,OMODE
        MOV     DL,12
        INT     21H                     ;Output form feed if printer
DOSRET: RET                             ;No file found, so end program
FOK:    MOV     DX,OFFSET FNAM          ;Point to file name
        MOV     AX,3D00H
        INT     21H                     ;DOS Open for input
        JC      FAIL                    ;Terminate on file open error
        MOV     BX,AX                   ;Save file handle
        MOV     HAND,AX
        CALL    DOMASK                  ;Search source for match on mask
        JMP     BIGLP                   ;Go for next file
SRCHDSP ENDP
;
DOMASK  PROC    NEAR
RAGAIN: MOV     CNTX,0                  ;Clear continue switch
        MOV     AH,3FH                  ;DOS Read file
        MOV     DX,OFFSET BUF           ;Point to input buffer
        MOV     CX,40000                ;Set for maximum read
        INT     21H
        CMP     AX,CX                   ;See how much actually read
        JNE     NORM                    ;Input is < 40K bytes
        MOV     CNTX,-1                 ;Set "continue" switch
NORM:   MOV     SI,OFFSET BUF           ;Point to start of file
        MOV     CX,AX                   ;Save read length
DOMLP:  CALL    MATCH                   ;See if match on mask
        INC     SI
        TEST    SKIP,-1                 ;See if matched
        JZ      DOLP                    ;No
        CALL    SKPLNE                  ;Yes, so skip remainder of line
DOLP:   LOOP    DOMLP                   ;Loop through source
        TEST    CNTX,-1                 ;See if not at end of file
        JNZ     RAGAIN                  ;Not, so read again
        MOV     AH,3EH                  ;Close file
        INT     21H
        RET
DOMASK  ENDP
;
MATCH   PROC    NEAR
        PUSH    SI                      ;Save file pointer
        PUSH    CX                      ;  and count
        LODSB                           ;Get first data byte at current loc
        MOV     CX,MCNT                 ;Load count of masks
        JCXZ    MAQUIT                  ;No masks, so exit
        MOV     DI,OFFSET XMASK         ;Point to mask save area
MALP:   CMP     AL,BYTE PTR [DI+1]      ;Match on first byte of mask
        JNE     MASTP                   ;No match
        CALL    STRING                  ;Match full string against mask
        JNC     MAQUIT                  ;Hit
MASTP:  ADD     DI,21                   ;Step to next mask
        LOOP    MALP                    ;Loop through all masks
MAQUIT: POP     CX                      ;Exit-No carry = match, Carry=no match
        POP     SI
        RET
MATCH   ENDP
;
STRING  PROC    NEAR
        PUSH    DI
        PUSH    CX
        PUSH    SI
        DEC     SI                      ;Step back to current loc
        MOV     CL,BYTE PTR [DI]        ;Get mask length
        INC     DI                      ;Point to mask
        XOR     CH,CH
```


PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

```
        REPE    CMPSB                   ;Compare mask with string
        STC                             ;Set not matched flag
        JNE     NOFND                   ;Take not matched exit
        CALL    PRINT                   ;Exact match, so output
        CLC                             ;Set match flag
NOFND:  POP     SI                      ;Exit
        POP     CX
        POP     DI
        RET
STRING  ENDP
;
GETMSK  PROC    NEAR
        PUSH    DI
        PUSH    CX
        MOV     DX,OFFSET IBUF          ;Set for DOS string input
        MOV     AH,10
        INT     21H
        MOV     CL,ILEN                 ;Get length of input string
        XOR     CH,CH
        JCXZ    GMQUIT                  ;Zero, so end of masks
        MOV     SI,OFFSET IDAT          ;Point to mask
        MOV     AL,CL                   ;Length of mask
        STOSB                           ;Save length
        REP     MOVSB                   ;   and mask
        INC     MCNT                    ;Step mask count
        CLC                             ;Normal exit flag
GMRET:  POP     CX
        POP     DI
        RET
GMQUIT: STC                             ;Set end of masks flag
        JMP     GMRET
;
PRINT   PROC    NEAR
        INC     LCNT                    ;Step line count
        MOV     AL,LCNT                 ;Load line count
        CMP     AL,LMAX                 ;See if over "page"
        JBE     PRTGO                   ;No, so continue
        MOV     LCNT,0                  ;Clear line count
        CMP     MODE,'P'                ;See if printer mode
        JE      PTT                     ;Yes
        MOV     DX,OFFSET WKSG
        MOV     AH,9
        INT     21H                     ;Output "wait for key message"
        MOV     AH,8
        INT     21H                     ;Wait for key stroke
        MOV     DX,OFFSET CRLF
        MOV     AH,9
        INT     21H                     ;Restore cursor
        JMP     SHORT PRTGO
PTT:    MOV     DL,12
        MOV     AH,OMODE
        INT     21H                     ;Output forms feed
PRTGO:  MOV     SKIP,-1                 ;Set skip flag for this line
        MOV     CX,SI
        SUB     CX,OFFSET BUF           ;Get length of line
        STD
        MOV     AL,10
        MOV     DI,SI
        REPNE   SCASB                   ;Reverse scan for end of previous line
        CLD
        ADD     DI,2                    ;Point to start of this line
        MOV     SI,OFFSET FNAM          ;Point to source file name
        MOV     CX,14                   ;Set length of name field
        MOV     AH,OMODE                ;Set for correct output media
PRTFIL: LODSB                           ;Load file name byte
        OR      AL,AL                   ;Null marks end of name
        JE      PAD
        MOV     DL,AL
        INT     21H                     ;Output name byte
        LOOP    PRTFIL                  ;Loop through name
        JMP     SHORT NOPAD
PAD:    MOV     DL,' '                  ;Pad out file name to 14 characters
PBLK:   INT     21H                     ;   with blanks
        LOOP    PBLK                    ;Loop through padding
NOPAD:  MOV     CX,65                   ;Set for max line of 79 bytes
        MOV     SI,DI                   ;Point to start of source line
DLP:    LODSB                           ;Load source line byte
        MOV     DL,AL
        INT     21H                     ;  and output
        CMP     AL,10                   ;See if its end of line (LF)
        JE      DATDNE                  ;Yes
        CMP     AL,1AH                  ;See if its end of file (EOF)
        JE      DATEND                  ;Yes
        LOOP    DLP                     ;Loop through source line
DATEND: MOV     DL,13                   ;Output CR
        INT     21H
        MOV     DL,10                   ;Output LF
        INT     21H                     ;End of source file exit
DATDNE: RET                             ;End of line Exit
PRINT   ENDP
;
SKPLNE  PROC    NEAR
        MOV     DI,SI                   ;Point to current line position
        MOV     AL,10
        REPNE   SCASB                   ;Forward scan to end of line
        MOV     SI,DI                   ;Make current ptr, start of next line
        MOV     SKIP,0                  ;Clear line skip flag
        INC     CX                      ;Adjust counter
        RET
SKPLNE  ENDP
;
CODE    ENDS
;
        END     SRCHDSP
```

PROGRAMMER'S JOURNAL

Famous Cartoons Reformatted as T-Shirts

See inside back cover for details.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

An Assembler Programming Suggestion

by Graham Pearson

If you're in the business of writing programmer utilities, you are probably writing code in Assembler and distributing your software as one or more .OBJ files, ready for linking with your clients' modules. Invariably, your clients are using a high-level language rather than Assembler, so your Assembler code is the result of hours of studying a compiler manual. If you're lucky, this manual has a chapter entitled "Interfacing with Assembler Code," so your utility has a good chance of working the first time (second time. . . third time. . .). More than likely, you rely on DEBUG to step you through some trial runs and you end up writing your own chapter on interfacing protocol.

But there are a few brave souls out there who are writing application systems in Assembler and who might like to use your utilities. So perhaps you should provide an Assembler interface. This article suggests a coding scheme that makes it easy for one person's assembly code to incorporate another person's assembly code.

The Culprits: MOV OFFSET and LEA

If you're not careful, assembly code written by two different programmers comes undone at LINK time. Problems are usually avoided if "far calls" are used to each other's code. But this convention precludes the resulting .EXE file from being converted to a .COM file.

Problems are introduced with "near calls" unless a compatible segment naming convention is adopted by both programmers. Control transfer instructions such as JMP and CALL are immune at LINK time because they transfer control to relative addresses rather than absolute ones. The two instructions which cause most havoc at LINK time are

```
MOV     register, OFFSET label (or variable)
LEA     register, label (or variable)$$
```

To illustrate this problem, let's look at a simple example. The USER procedure listed below represents code being used to call your BLACKBOX procedure, whose source is listed second. Remember that the BLACKBOX source is not available to your user because you provide only the .OBJ file (for several reasons!).

```
CSEG            SEGMENT BYTE PUBLIC 'CODE'
                ASSUME  CS:CSEG
                EXTRN   BLACKBOX:NEAR

                ORG     100H        ;Setup for .COM program

USER            PROC    FAR
        MOV     DX,OFFSET USER_MSG
        MOV     AH,9                ;Use DOS to print a string
        INT     21H                 ;terminated by a $ symbol

        CALL    BLACKBOX            ;Call third-party routine

        INT     20H                 ;Return to DOS
USER            ENDP

USER_MSG        DB      0DH,0AH,'USER message',0DH,0AH,'$'

CSEG             ENDS
END              USER               ;Define entry point
```

```
CODE            SEGMENT BYTE PUBLIC 'CODE'
                ASSUME  CS:CODE
                PUBLIC  BLACKBOX

BLACKBOX        PROC    NEAR
        MOV     DX,OFFSET BBOX_MSG
        MOV     AH,9
        INT     21H
        RET                         ;Return to USER
BLACKBOX        ENDP

BBOX_MSG    DB  0DH,0AH,'THIRD-PARTY message',0DH,0AH,'$'

CODE            ENDS
END
```

If the USER procedure is assembled, linked with BLACKBOX.OBJ, converted to a .COM file, and then run, your client sees his 'USER message' appear but is then rudely awakened by smiley faces and beeps when the program tries to display the 'THIRD- PARTY message'. A disassembly of the .COM file, courtesy of DEBUG, yields the following:

```
C>DEBUG USER.COM

-u 100 10b
1234:0100 BA0C01        MOV     DX,010C
1234:0103 B409          MOV     AH,09
1234:0105 CD21          INT     21
1234:0107 E81300        CALL    011D
1234:010A CD20          INT     20

-d 10c 11c
1234:0100                ....                ..US
1234:0110  ............        ER message..$

-u 11d 124
1234:011D BA1500        MOV     DX,0015          <------
1234:0120 B409          MOV     AH,09
1234:0122 CD21          INT     21
1234:0124 C3            RET

-d 125 13c
1234:0120       ............            ..THIRD-PAR
1234:0130 ............          TY message..$
```

The offending instruction is indicated by the <------ symbol. The MOV OFFSET instruction is not adjusted to the absolute address of BBOX_MSG at LINK time. It continues to reflect the relative offset of BBOX_MSG from the start of the BLACKBOX procedure. The same result is obtained if we use the LEA instruction rather than MOV OFFSET; the user is treated to the delights of the program segment prefix (PSP) rather than your THIRD-PARTY message.

A Heavy-Handed Solution

An easy way out of this predicament for you, the supplier of the BLACKBOX procedure, is to stipulate that a specific code segment name be used. In the above example, if the USER PROC is declared within the code segment:

```
CODE    SEGMENT BYTE PUBLIC 'CODE'
```

both code segments now have the same name and all is well at LINK time. However, this is an unnecessary and heavy-handed approach. A little more effort on your part can make it a lot easier for your user.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

GROUP to the Rescue

The GROUP statement causes LINK to collect separately assembled segments of code into one logical group. The addresses of all labels and variables are adjusted so that they are relative to the beginning of the logical group rather than to the beginning of the segment in which they are defined.

This is exactly what we need to rectify the phenomenon above. If we incorporate a GROUP statement in our BLACKBOX procedure, the calling procedure has two options: Either it must use the same segment naming convention as we do (the heavy-handed approach), or it can include its own GROUP statement.

LEA rather than MOV OFFSET

Our BLACKBOX procedure needs one final modification. LINK must be made aware of each MOV OFFSET instruction by including the GROUP override operator:

```
MOV     register, OFFSET groupname:label
```

This makes the long-winded MOV OFFSET instruction even more cumbersome. A more succinct approach is to use the LEA instruction, which takes into account a GROUP offset if we change our ASSUME CS: statement to reflect the name of the group rather than the name of our local code segment.

The modified source codes are listed below:

```
; The following statements are necessary ONLY if the user's
; code segment has a name other than CODE.
; The dummy segments are defined so that the GROUP statement
; correctly assembles and so that LINK has no doubt in which
; order the separate code segments are to be linked.

CGROUP          GROUP   CSEG, CODE

CSEG            SEGMENT BYTE PUBLIC 'CODE'
CSEG            ENDS

CODE            SEGMENT BYTE PUBLIC 'CODE'
CODE            ENDS


; User's unmodified code starts here:

CSEG            SEGMENT BYTE PUBLIC 'CODE'
                ASSUME  CS:CSEG
                EXTRN   BLACKBOX:NEAR

                ORG     100H        ;Setup for .COM program

USER            PROC    FAR
        MOV     DX,OFFSET USER_MSG
        MOV     AH,9                ;Use DOS to print a string
        INT     21H                 ;terminated by a $ symbol

        CALL    BLACKBOX            ;Call third-party routine

        INT     20H                 ;Return to DOS
USER            ENDP

USER_MSG        DB      0DH,0AH,'USER message',0DH,0AH,'$'

CSEG            ENDS
END             USER                ;Define entry point

------------------------------------------------------------

; Our GROUP statement reflects the name of our code segment:

CGROUP          GROUP   CODE


CODE            SEGMENT BYTE PUBLIC 'CODE'
                ASSUME  CS:CGROUP           ; <-- Note change
                PUBLIC  BLACKBOX

BLACKBOX        PROC    NEAR

        MOV     DX,OFFSET CGROUP:BBOX_MSG   ; <-- Either this
        LEA     DX,BBOX_MSG                 ; <-- or this

        MOV     AH,9
        INT     21H
        RET                         ;Return to USER
BLACKBOX        ENDP

BBOX_MSG    DB  0DH,0AH,'THIRD-PARTY message',0DH,0AH,'$'

CODE            ENDS
END
```

A disassembly of the resulting .COM file yields the following:

```
C>DEBUG USER.COM

-u 100 10b
1234:0100 BA0C01        MOV     DX,010C
1234:0103 B409          MOV     AH,09
1234:0105 CD21          INT     21
1234:0107 E81300        CALL    011D
1234:010A CD20          INT     20

-d 10c 11c
1234:0100                ....                ..US
1234:0110  .............        ER message..$

-u 11d 128
1234:011D BA2901        MOV     DX,0129          <------
1234:0120 8D162901      LEA     DX,[0129]        <------
1234:0124 B409          MOV     AH,09
1234:0126 CD21          INT     21
1234:0128 C3            RET

-d 129 140
1234:0120           .......             ..THIRD
1234:0130  .................    -PARTY message..
1234:0140  .                    $
```

Note that both the MOV OFFSET and LEA instructions correctly reflect the absolute address of 'THIRD-PARTY message'.

Summary of Recommendations

Although the examples above illustrate the use of a GROUP directive with the CODE segment, the same can be applied to the DATA or STACK segments.

The following guidelines make it easier for other Assembler programmers to correctly incorporate your assembly routines with theirs, particularly if yours is a NEAR procedure:

1. Include a GROUP directive for each of your segments at the top of your source code:

```
CGROUP          GROUP   CODE
DGROUP          GROUP   DATA
SGROUP          GROUP   STACK
```

2. Change all ASSUME directives to reflect the appropriate GROUP name:

```
ASSUME  CS:CGROUP
ASSUME  DS:DGROUP
ASSUME  SS:SGROUP
```

3. For the sake of brevity, use the LEA instruction rather than the MOV OFFSET GROUP: instruction. Compare the following equivalent statements:

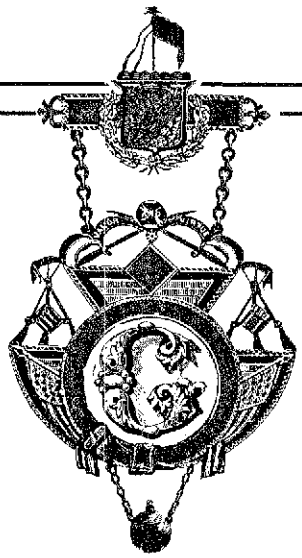
```
LEA     DX,MESSAGE
MOV     DX,OFFSET CGROUP:MESSAGE
```

4. Tell your Assembler clients they have one of two options:

   a. Name their segments the same as yours.
   b. Include the appropriate GROUP statements at the top of their source code:

```
CGROUP  GROUP   USER_CODESEG, YOUR_CODESEG

USER_CODESEG    SEGMENT BYTE PUBLIC 'CODE'
USER_CODESEG    ENDS

YOUR_CODESEG    SEGMENT BYTE PUBLIC 'CODE'
YOUR_CODESEG    ENDS
```


PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED



The Proposed ANSI C Language Standard, Part 3

by Rex Jaeschke

©Rex Jaeschke, 1986. All rights reserved.

#include and fopen

In my last column we covered the Preprocessor in detail. I would like to revisit the **#include** directive briefly, to address a DOS-specific concern that was raised in committee on a number of occasions. It has to do with DOS' method of naming disk subdirectories.

The completely qualified name of a DOS disk file has the following format:

\dir1\dir2\ ... \filename.ext

This same format is used by UNIX systems, but UNIX uses the forward slash / to separate directory levels whereas DOS uses the backslash \. The problem arises with directory and filesnames such as

\totals\accum.dat
\summary\file1.sum
\times\filecopy.txt

Here, either the directory or the filename (or both) begins with the sequence **\x** where **x** is a recognizable special character sequence within a C literal string. The problem then becomes, "How do I name this file when accessing it?" The following example provides the solution for accessing both a header and a data file which reside on such a directory path.

```
#include <stdio.h>
#include "\t\f\test.h"

main()
{
        FILE *infile;
        int c;

        if ((infile = fopen("\\t\\f\\test.dat","r")) == NULL) {
                printf("Can't open input file\n");
                exit(1);
        }
        ...
}
```

The first argument to **fopen** is a pointer to **char**, which in this case is a literal string. To ensure that the **\t** and **\f** are not incorrectly interpreted as the tab and form-feed characters, respectively, prefix each character sequence with a \. This causes the sequence \\ to be interpreted as a literal backslash and *not* as a character sequence prefix. That's nothing new; however, the header file **test.h** does not require this. Why not?

The Standard does *not* consider the text between ". . ." in an **#include** directive to be a C language literal string. It is treated merely as a sequence of characters whose meaning is implementation-defined. So while it looks just like a literal string, it is not subject to the same translation that the compiler applies to literal strings within C language statements.

This approach may already be supported by your DOS compiler although there have been discrepancies in the past.

There is a potential trap with "conforming" compilers and the runtime library functions that accept a filename as an argument. Two new character sequences have been added to the language: **\a** represents alert, which could cause the terminal bell or buzzer to sound; **\v** represents the vertical tab. Also, the sequence **\xddd** allows a character bit-pattern to be specified using from one to three hexadecimal digits.

The significance of this is that the new sequences will cause runtime errors if existing code contains something like

```
fopen("\accounts.dat","r")) ...
fopen("\volume.dat","r")) ...
fopen("\xrays.txt","r")) ...
```

All such file references will need to be changed to

```
fopen("\\accounts.dat","r")) ...
fopen("\\volume.dat","r")) ...
fopen("\\xrays.txt","r")) ...
```

The Standard functions that use filenames, besides **fopen**, are **remove**, **rename**, and **freopen**.

The existing octal sequence specifier remains as **\ddd** where **ddd** contains from one to three octal digits. The relevance of this is that to reference filenames that begin with a sequence of decimal digits, you will also need an extra backslash as in

```
fopen("\\860501.dat","r")
```

Note that if the extra backslash were omitted, you would have an invalid octal constant. If the filename were "**\123.dat**", the error would not be so obvious. A good rule of thumb is to *always* use \\ in path names instead of \, even when the character following the \ is not a recognizable sequence suffix.

Trigraphs

Except in exceptional circumstances, every DOS computer is sold with a keyboard for all of the printable characters understood by C. However, it is possible to use a "foreign" terminal that does *not* have all these characters on a DOS computer. Such terminals are older and/or cheaper; a popular example is the ADM-3. Note also, that many European keyboards are based on the ISO 646-1983 Invariant Code Set, which is a subset of the seven-bit ASCII code set. This set is missing several critical C characters.

To enable these missing characters to be represented in source code files, a set of trigraph sequences has been added to C. Each sequence contains three characters of which the first

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

two are ??, and the Standard contains the following statement: "The following trigraph sequences in the source program shall be replaced by the specified characters in the source character set

trigraph	meaning
??=	#
??(	[
??/	\
??)	]
??'	^
??<	{
?? !	|
??>	}
??-	~

Any other sequence of characters that begins with ?? is unchanged."

Since (almost) all of you have the full ANSI keyboard, why is this of interest? Because a conforming compiler must include support for trigraphs, and such support could accidentally (although unlikely) change the behavior of existing code that contains literals with sequences of question marks. For example, if your programs contain literals such as

```
printf("??(Problem here)\n");
printf("??<message>\n");
printf("??'text'\n");
```

they will be interpreted as

```
printf("[Problem here)\n");
printf("{message>\n");
printf("^text'\n");
```

which is certainly not what you would want. To correct this problem, modify the strings:

```
printf("\??(Problem here)\n");
printf("\??<message>\n");
printf("\??'text'\n");
```

The new character sequence \? allows a question mark to be treated as a literal rather than as the start of a trigraph sequence as demonstrated. In cases where a question mark's use need not be so differentiated, either \? or ? can be used. For example, the expressions

```
        if (c == '?') ...
and     if (c == '\?') ...
```

are equivalent.

In future columns I'll discuss various new DOS compilers and interpreters and lint programs that understand something about the Proposed Standard, as well as continue with tutorial material.

In September, Lattice will host the X3J11 meeting near Chicago, Illinois, and in December, Plum Hall will host the meeting in Atlantic City, New Jersey. □

Rex Jaeschke is an independent computer consultant, writer, and lecturer. He is also a member of the X3J11 Committee and co-founder and Editor of the C Journal.

If you would like to comment on or question any aspect of the Proposed ANSI C Standard, write to Rex Jaeschke at

2051 Swans Neck Way
Reston, VA 22091

Rex will try to address your concerns in future columns as space permits.

DeSmet C

now with 32-Bit Pointer Option

C88 ..... *still* **$109**

The editors' choice for fast compilation and execution. The **price/performance winner** in all major C benchmarks since 1983. Includes Compiler, Assembler, Binder, Librarian, Execution Profiler and Full Screen Editor. Supports both disk and memory resident Overlays. Contains both 8087 and Software floating point support. Full STDIO library.

Large Case Option **$50**

Makes a great C Compiler even better. Adds 32-Bit Pointers to C88 so you can utilize all of your PC. Groups scalar and static data for fast access. Supports the D88 debugger.

D88 ..... **$50**

Gain most of the benefits of an interpreter while losing none of the run-time speed of the C88 compiler. Display C source and variable contents during execution. Set breakpoints by function name or line number. Examine and set variables by name using C expressions.

order direct from:

C Ware Corporation
505 W. Olive, Suite 767, Sunnyvale, CA 94086 U.S.A.
(408) 720-9696 — Telex: 358185
We accept VISA, MasterCard & American Express

Circle Number 38 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Monitoring Stack Usage and a Guide to Startup Source Code

by William James Hunt

Monitoring Stack Usage

It is often necessary to find out how much stack space a C program requires. While we can calculate the amount of memory needed by examining C source files, it can be a lot of trouble. For C library functions whose source code is not available, we cannot find out their requirements. The best solution is to monitor the stack usage of an executing program. With the DOS debugger or a symbolic debugger such as Lattice's C-Sprite, you can set breakpoints at key points in your program and then examine register values to determine stack usage. But why not let the computer check stack usage and display the results? The monstk.c source file below contains C functions that measure stack usage for programs written with Lattice C.

The top level function stk__use displays the amount of stack space currently in use and the maximum used since execution started. The functions in monstk.c assume that the stack grows downward and can reach a maximum size of 64 Kbytes. These assumptions are valid for Lattice and Microsoft C on computers with an Intel 8088 family processor. The low-level details to complete the stack monitoring are contained in the monstkl.c source file (for Lattice Version 2.10 or later) and monstkm.c (for Microsoft 3.0 or IBM C). For the Microsoft version, the microstk.asm file is also needed. The Microsoft version works with small, medium, and large models; the Lattice versions work with the S,P,D and L models. (Use the -MLS and -MDS options when compiling monstk.c and monstkl.c.)

Both monstkl.c and monstkm.c use some undocumented information about stack placement and size. I got that information by examining the stack checking code generated by the two compilers and the startup code. For Lattice C, the startup source code is provided with the compiler; for Microsoft 3.0, I used the debugger to disassemble the startup code. (The source code shown was tested with Lattice Versions 2.15 and 3.00 but should work with Versions 2.10–2.14 as well.)

These stack monitoring functions are not exact. Since the cumulative use test looks for words in which a pattern is altered, it can't detect local variables that are allocated but never set. All the functions use the stack themselves even though scratch data is declared as static storage to minimize interference. The top level function, stk__use, calls printf, which uses about 300 bytes of stack space itself. When an uncertainty of 300 bytes is too much, you can call stk__now and stk__max directly and store the values for later printing.

The teststk.c file illustrates the use of the monitoring functions. A sample output using Lattice C is shown below.

```
C>teststk =6000
 in main          - STACK:    212 Bytes NOW     226 MAX
 in Funl - n =  3 - STACK:    720 Bytes NOW    1056 MAX
 in Funl - n =  2 - STACK:   1228 Bytes NOW    1564 MAX
 in Funl - n =  1 - STACK:   1736 Bytes NOW    2072 MAX
 in Funl - n =  0 - STACK:   2244 Bytes NOW    2580 MAX
 in Funl - n =  1 - STACK:   1736 Bytes NOW    2596 MAX
 in Funl - n =  2 - STACK:   1228 Bytes NOW    2596 MAX
 in Funl - n =  3 - STACK:    720 Bytes NOW    2596 MAX
 back in main     - STACK:    212 Bytes NOW    2596 MAX
```

```
/* monstk.c - monitor stack usage */
#include "stdio.h"
static int *pmark ;          /* use for storing/checking pattern */
static long diff ;           /* scratch counter */
static int *pmax ;           /* records last location changed */
#define PATTERN 0x1234       /* store this distinctive pattern */

long stk_bytes() ;           /* calculate stack use given a pointer */
long stk_size() ;            /* calculate stack size */

long stk_now()               /* measure current stack use */
 {
   int marker ;

   return( stk_bytes(& marker) ) ;
 }


stk_mark()                   /* fill unused stack with a pattern */
 {
   int marker ;

   pmark = & marker ;        /* start at our marker */
   diff = stk_size() - stk_bytes( & marker) ;
   while( diff >= 0 )        /* store the pattern in rest of stack */
        { *pmark = PATTERN ;
          pmark -- ;
          diff = diff - sizeof(int) ;
        }
    }


long stk_max()               /* measure cumulative stack use */
 {
   int marker ;

   pmark = & marker ;        /* start here */
   diff = stk_size() - stk_bytes( & marker) ;
   while( diff >= 0 )        /* look for altered words to bottom */
        { if( *pmark != PATTERN)
                pmax =  pmark ;
          pmark -- ;
          diff = diff - sizeof(int) ;
        }
   return( stk_bytes(pmax) ) ;
 }


stk_use(where)               /* display current/cum. stack use */
 char where[] ;              /* describes where called from */
 {
   printf(" %-20s - STACK: %6ld Bytes NOW   %6ld MAX \n", where,
        stk_now() , stk_max() );
 }


/* monstkl.c - low level stack usage functions - Lattice Version */
/* works with the small(S), program(P), data(D) and large(L) */
/* memory models. */
/*  With the L or D models, use the -S option (-MLS or -MDS) */
extern unsigned _top ;       /* top of stack */
extern unsigned _stack ;     /* stack size constant */

long stk_bytes(p)            /* calculate stack use to */
 int *p ;                    /* this address */
 {
   return(  ( (long) _top ) - ( ((long) p) & 0xffffL) ) ;
 }


long stk_size()              /* get stack size */
 {
   return( (long) _stack ) ;
 }


/* monstkm.c - low level stack usage functions - Microsoft 3.0 */
extern unsigned _atopsp ;    /* top of stack */
extern unsigned *pstklim ;   /* points to stack limit */

long stk_bytes(p)            /* calculate stack use to */
 int *p ;                    /* this address */
 {
   return(  ( (long) _atopsp ) - ( ((long) p) & 0xffff) ) ;
 }


long stk_size()                    /* get size of stack */
 {
   return(  (long) ( _atopsp - *pstklim ) ) ;
 }


;  MICROSTK.ASM - Microsoft 3.0 stack usage support
;  define pointer to STKHQQ
DGROUP  group   _DATA
_DATA   segment  word public 'DATA'
   assume    ds:DGROUP

   extrn     STKHQQ:WORD
   public    _pstklim
_pstklim dw  OFFSET DGROUP:STKHQQ
   DW    DGROUP
_DATA   ends
   END
```


PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

```
/* teststk.c - demonstrate stack monitor functions */
#include "stdio.h"

main()
 {
   stk_mark() ;              /* mark stack */
   stk_use("in main");       /* show usage before */
   funl(3) ;                 /*      recursion */
   stk_use("back in main") ;/* and after */
 }


funl(n)
 int n ;
 {
   char x[500] ;
                             /* build a display message */
   sprintf(x,"in Funl - n = %2d",n) ;
   stk_use(x) ;              /* display stack use before */
   if( n > 0 )
        { funl(n-1) ;        /*      recursive call */
            stk_use(x) ;     /* and after return */
        }
 }
```

A Guide to Startup Source Code

One of the advantages of C for serious software development is its open, modular structure. To take full use of that openness, you need to understand the environment in which C programs operate. Lattice C provides source code for the startup sequence executed before your main function is called. This is an overview of what that startup code does.

When DOS runs an .EXE file compiled with Lattice C, execution begins at the symbol C in the object file C.OBJ. (There is a version of C.OBJ for each memory model.) The corresponding source file, C.ASM, covers all memory models. The include file DOS.MAC defines the memory model when C.ASM is assembled. The startup sequence in C.OBJ calls the function __main in the __MAIN.OBJ library module. The corresponding source file is __MAIN.C. Finally, __main calls our function main.

```
DOS  =>   C        =>    _main     =>    main
        in C.ASM         in _main.c       in our source file
```

These two files, C.ASM and __MAIN.C, carry out the following steps:

C.ASM:

1. Sets up the 8086 segment registers—DS immediately, SS after the stack's position has been established, and ES just before calling __main.
2. Gets information from the program segment prefix that DOS places in front of the loaded program. This information includes the amount of memory defined (up to 640 K for example), the location of the DOS environment string, and the command line typed after the program name.
3. Scans the command line for a stack size specification (=10000 for example) and file redirection information. Under DOS 2.x or 3.x, command.com would have removed the redirection information. With Lattice 3.00, a minimum heap size can also be specified.
4. Computes the stack size to use in the program. The C.ASM file declares a small stack (128 bytes) segment, but the startup code immediately relocates the stack to a larger area. The size of the final stack is determined by several inputs. The initial value of the __stack variable may be replaced by the =xxxx stack size on the command line. The resulting __stack value is used unless it is less than the minimum stack size (512 bytes in version 2.15). If the memory size in the program segment prefix is too small for code, static data, and the stack, the program will display an error message and exit to DOS.
5. Locates the dynamic allocation area (heap). A minimum amount of memory is reserved and the rest returned to DOS via the Memory Allocation call 0x4A. If there is not enough room for this heap size, the program displays an error message and exits.
6. Creates a pointer to the environment string. (For memory models with 16 bit pointers, the environment data is copied to the stack so that it can be addressed via a 16 pointer.)
7. The command line is copied to the stack. The stack size (=xxxxx) and I/O redirection information is not copied.
8. Checks for and initializes an 8087 or 80287 math chip.
9. Calls the __main function.

Continued . . .

C CODE FOR THE PC

source code, of course

GraphiC 3.0 hi-res color plots	$300
QC88 C Compiler	$90
Concurrent C	$45
Coder's Prolog in C	$45
LEX	$25
YACC & PREP	$25
Small-C compiler for 8088	$20
tiny-c interpreter & shell	$20
Xlisp 1.5a & tiny-Prolog	$20
C Tools	$15

The Austin Code Works
11100 Leafwood Lane
Austin, Texas 78750-3409
(512) 258-0785

Free shipping on prepaid orders — No credit cards

Circle Number 30 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

_MAIN.C:

10. Breaks the command line into "words" replacing blanks by '\0' characters to make C strings. It creates the argc and argv arguments expected by main.

11. Prepares the FILE structures for stdin, stdout, and stderr standard I/O files.

12. Calls the main function. When main returns, it calls the exit function (or _exit if it was compiled with the TINY option). The C.ASM file also contains code for a normal exit to DOS, an error exit, and a stack overflow error handler.

This list of steps applies to Versions 2.15 and 3.00 of Lattice C, but the steps in earlier versions of Lattice C would be similar. Since startup source code was not available for Microsoft 3.0, this applies only to Lattice C.

The source files for these stack monitoring functions and for the C memory resident functions in the July/August 1986, issue of *Programmer's Journal* are available on an IBM PC format floppy disk (360 Kbyte, 5 and¼ inch)—along with other C source code—from

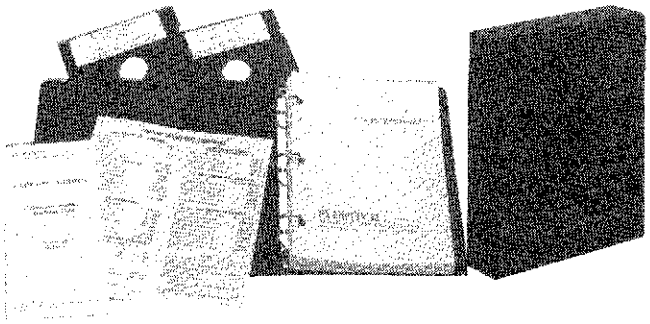
William James Hunt
Lattice C Users' Group
P.O. Box 271965
Concord, CA 94527

Enclose a check or money order for $20.00 ($15.00 for Lattice C Users' Group members) payable to Lattice C Users' Group. Add $3.00 for delivery outside the USA. □

Bill Hunt writes the Lattice C Users' Group *newsletter, where this material first appeared, and is also the author of* The C Toolbox, *published by Addison-Wesley.*

To write interactive programs, create graphics and prepare demos for IBM or Texas Instruments computers, specify

C-DISPLAY UTILITIES™



C-DISPLAY COMPILER™ $125

Helps you design the screens interactively, call them from your programs and use them in demos and for prototyping.
Bonus: Free tutorial on display management with every order.
Option: DemoExpress Viewer,™ $29, to create demos, tutorials and continuous presentations without writing a single line of code.

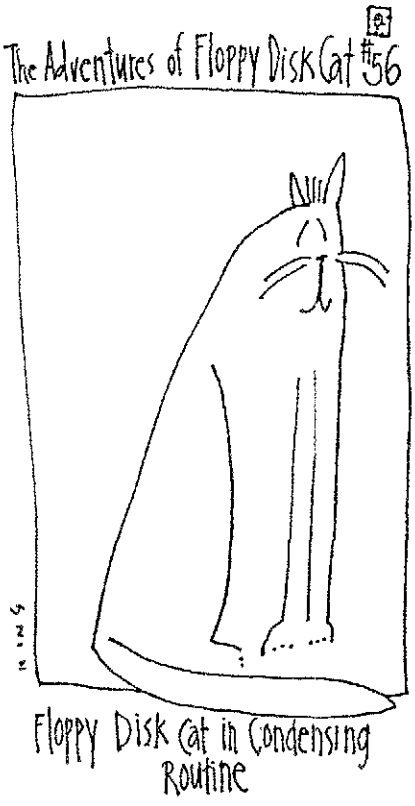
C-DISPLAY LIBRARIAN™ $145

Makes it easy to write interactive programs. It supports BIOS video functions, screen editing, data entry and validation, bit mapped graphics, and much more. Compatible with Lattice C and Microsoft C 3.0, Fortran 3.3 and Pascal 3.3 compilers.

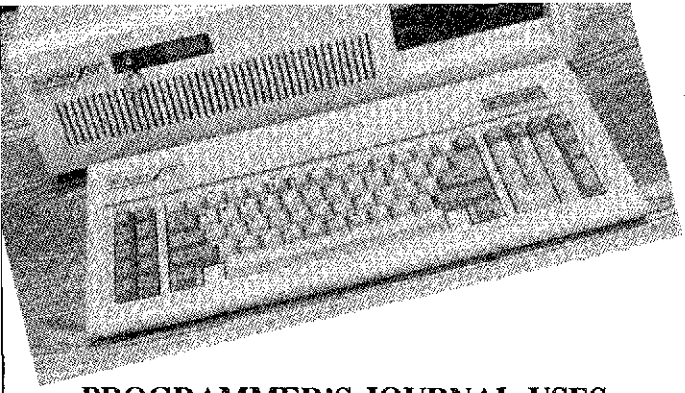
Both units available for immediate shipment.
By mail or write for the name of the dealer nearest you.

SYDETECH
SYSTEM DEVELOPMENT TECHNOLOGIES
43-23 Colden St., Suite 17C, Flushing, NY 11355, (718) 886-0221

Circle Number 74 on the Reader Service Card



PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

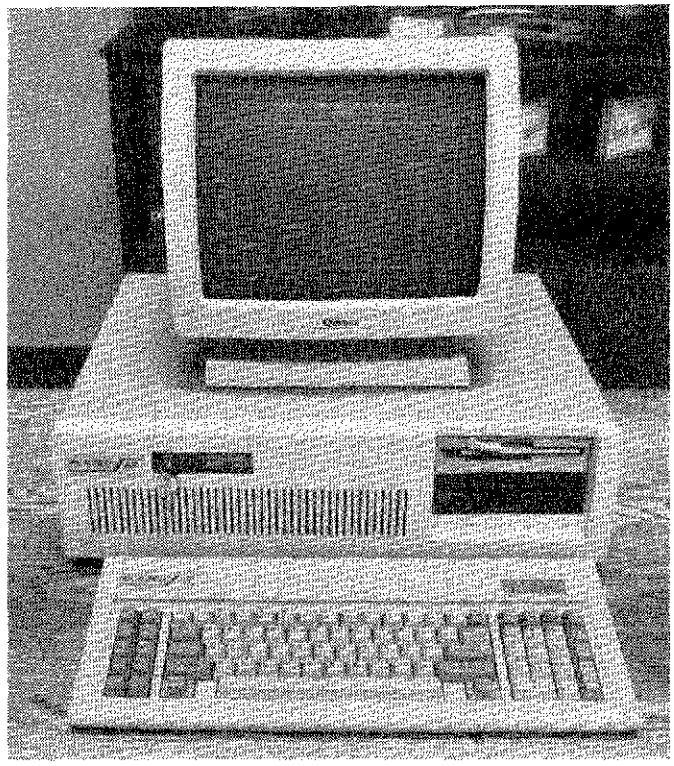


BLAZER XT&AT

PROGRAMMER'S JOURNAL USES HARD WORKING BLAZERS.

Every day the staff at **Programmer's Journal** uses dependable, hard working **BLAZERS** to take care of a wide variety of urgent daily business — from office correspondance and accounting needs to copy editing and testing software and hardware for review.

When reliability under deadline pressure counts... put a BLAZER to work for you!



BASIC BLAZER XT

- IBM PC/XT ™ Compatible
- 640K Dynamic RAM
- 8 Slot Motherboard
- Color or High Res Mono Graphics Card
- Dual 5-1/4" 360K Disk Drives
- 135W Power Supply
- Parallel Printer Port
- Improved Keyboard
- Hi Res Monochrome Monitor
- $1,295.00

CUSTOM CONFIGURATIONS

AT & XT accessories include:
1.2 Meg Floppy Drives,10, 20 & 30 Meg Hard Drives, Parallel, Serial, MF I/O, Color & Mono Graphics Cards, Keyboards, Color Monitors, etc. Call for prices.

BASIC BLAZER AT

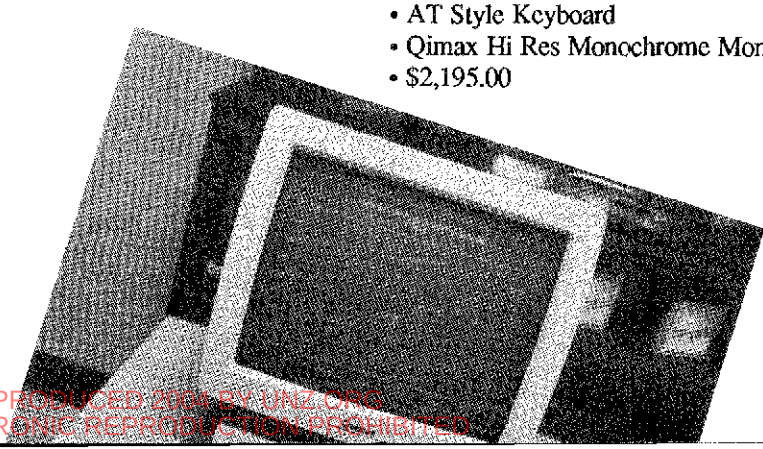
- IBM AT ™ Compatible
- Dual 6 & 8MHZ Clock Speed
- 1 Meg Dynamic RAM
- 8 Slot Motherboard
- Color or Hi Res Mono Graphics Card
- Dual 5-1/4" 360K Disk Drives
- 200W Power Supply
- Clock Calendar on Motherboard
- Parallel Printer Port
- AT Style Keyboard
- Qimax Hi Res Monochrome Monitor
- $2,195.00

PUT ONE TO WORK FOR YOU!

ECI

EMERALD COMPUTER INTERNATIONAL
540 Oak Street
Eugene, Oregon 97401
503/343-9393

Circle Number 76 on the Reader Service Card



PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Short Boards

Benchmarks were run, variously, on a stock IBM PC; a stock 6MHz IBM AT; and a Blazer 4.77MHz XT clone and Blazer 6/8MHz AT clone, both from Emerald Computer in Eugene.

Two issues ago (*PJ*, Vol. 4, No. 3) we reviewed three 80286 coprocessor boards for the PC. This issue, we examine two 80286 short board accelerators: one from PC Technologies and one from Orchid Technologies.

A difficulty in using an 80286 to speed up an 8088-based PC is that the 80286 has a 16-bit bus and the 8088 has an 8-bit bus.

One approach to solving this incompatibility is the 286 coprocessor board, which commonly consists of a full 286 bus with 1 to 4 Megs of no-wait-state RAM, and an 80286 running its own copy of DOS at 8MHz. Programs are loaded from disk onto the coprocessor board and run, and the 8088 is left running a caretaker program on the motherboard, communicating with the coprocessor through dual-ported RAM and tending to the hardware as an I/O slave of the 286.

This approach has its difficulties though—with "badly behaved" software and hardware, and with emulation of DOS and BIOS calls by the combination of the 286 and 8088. The coprocessor approach also requires more hardware. Nevertheless, the environment looks useful for software development.

The short board approach is different.

Each of this issue's short boards consists basically of an 80286 running at 7.155MHz, assisted by an 8K cache of fast 16-bit RAM and supporting logic. The motherboard 8088 is removed from the PC, the board plugs into the 8088 socket with a cable, and it loads a program in 8K chunks from the motherboard, running it without wait-states from the cache.

To software and hardware, the system looks more like a standard PC than does a coprocessor board, and the short board hardware is cheaper. A disadvantage is that code must be continually cached from the slow 8-bit-wide 8088 motherboard memory in 8K chunks while a program is running.

THE CONTENDERS:

286 Express Card
PC Technologies
$595.00

also OEMed as:

The SpeedPak 286
Victor Technologies

The Race Card
Mountain Computer

Hardware:
Short card 7.155MHz 80286 replacement for the 8088

Optional 4.8- or 8MHz 80287

8K of static RAM cache; jumper enabled or disabled

Power: 7.5 Watts, 10 Watts with 80287

Compatible with LAN and mainframe communication products and conforms to the EMS specification

Manual:
The manual is brief and to the point.

Installation:
To remove the 8088, find a friend with slender hands. After that, it's easy. The card cable is plugged into the empty 8088 socket, and the card goes in a nearby half slot. The pins on the end of the Express cable are sturdy and have gone in and out of the 8088 socket many times now without problem.

TinyTurbo 286
Orchid Technologies
$695.00

also available as part of:

Turbo EGA
$995

HARDWARE:
Short card 7.155MHz 80286 replacement for the 8088 that allows the 8088 to remain in the system

8K of 55 nanosecond static onboard 16-bit cache; cache can be jumper enabled or disabled

Power consumption: 7 Watts

Socket for 80287, jumper selectable for 5MHz or 8MHz

A switch on the backplane does a cold start and switches between the 80286 and the 8088.

One year warranty

Manual:
The manual is a straightforward 12-page booklet.

Installation:
Installation is simple although the connecting cable is not as long as the Express's, and isn't as easy to install. See the description of the Express installation above.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

The Turbo EGA

Hardware:

Orchid has put the TinyTurbo on a full length board in combination with four display protocols: an EGA, a CGA, a Hercules-style mono graphics adapter, and a monochrome adapter. The board goes in the first full-length slot, and a long cable runs over intervening boards to the 8088 socket. The board will reside in the same system with a CGA or MDA. To use an EGA, the PC must have ROM ICs with a copyright date of 10/27/82 or later. Software to check the ROM date is included.

The backplane has a switch to move between 286 and 8088 mode, as well as between composite, RGB, and mono outputs. No printer port.

BENCHMARKING THE SHORT BOARDS

All benchmark times are in seconds.

The Caches

First, a few benches of the boards' caching algorithms.

The companies called each algorithm "simple and straightforward." Each cache has 4K of RAM "flag field" or look-up table, and whenever the 286 puts an address on the bus, the flag field is consulted to see if the memory location is already present. If it is not, the cache is loaded.

An 8MHz Orchid 286e coprocessor board was used as the 286 baseline. The coprocessor may be complex, but if the bench code does no DOS or BIOS or I/O calls, the board provides a measure of straight 8MHz, no-wait-state, 286 execution time. The times in the cache benches were normalized to 10 for the 286e. Also included was a stock 6MHz AT, a Compaq 286 Portable II, and a stock 4.77MHz PC.

First, a code loop written to run entirely within the cache:

```
here:
    AAM
    AAM
     .
     .
     .
    AAM
    LOOP here
```

AAM was repeated 32 times per loop. The instruction ASCII Adjust for Multiplication involves only the AX register, no memory access. It takes 83 clocks on the 8088, which helps avoid uncertainties that could be introduced by a drain of the pre-fetch queue. The loop was executed 64K times, so the initial cache load time had a minimal effect.

Comparing the short and long board times should show the difference in clock speed plus any overhead associated with maintaining the cache. (Even if the cache doesn't need to be filled during this loop, the flag field probably still needs to be checked during each instruction fetch.)

AAM instruction

286e	286 Express	TinyTurbo	6MHz AT	Compaq Port II	8088
10.0	11.2	11.2	13.33	10.06	80.3

In this benchmark, the short boards run just as efficiently as the uncached 286. (With the difference in clock speeds, 11.2 should be the theoretical limit for the short boards.)

Next, a loop that writes a register to memory. The location written to was only a few bytes away from the loop and within the cache's limits, but there was a substantial slowdown.

MOV LOCAL__WORD,AX

286e	286 Express	TinyTurbo	6MHz AT	Compaq Port II	8088
10.0	25.0	25.0	15.62	11.48	54.4

MOV LOCAL__BYTE,AL

286e	286 Express	TinyTurbo	6MHz AT	Compaq Port II	8088
10.0	15.52	15.52	15.62	11.48	43.8

Dick Barton at PC Technologies said that his board always writes directly to the slow motherboard memory, never to the cache.

On the other hand, both short boards will repetitively read a memory location from the loop with no appreciable speed loss, whether the location of the word read is a few bytes or 20K away. A loop containing alternating MOV AX,LOCAL__WORD1 and MOV AX,LOCAL__WORD2 instructions runs just as fast, even when the two locations are separated by 20K. Dick Barton said that this is because the boards cache data memory as well as code and can cache individual locations independently.

We leave it to the reader to speculate on how the algorithm ensures that data memory values held in the cache are always up-to-date.

Likewise, an immediate load, MOV AX,1111h ran fast:

286e	286 Express	TinyTurbo	6MHz AT	Compaq Port II	8088
10.0	11.7	11.7	20.33	15.10	66.1

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Short Boards, 286s AND 386s

A conversation with Dick Barton, VP Engineering, PC Technologies.

PJ: What would you like to tell us about your company?

Dick Barton: We're about three years old. We have one other product, a multi-user board for the PC that carries a 286 and an 8088, which we first showed at Comdex three years ago. It was just us and Intel with the 286 then. Actually, a lot of the development work on the 286 was done on that board. Then we brought out the short board in December of last year.

PJ: We've seen the price of the short board discounted around the country.

Dick Barton: That depends on how aggressive the OEMs are being. Victor, for example, is very aggressive right now.

PJ: What kind of RAM is on the card?

Dick Barton: 55 ns of static RAM, which is not really fast RAM anymore. There's 8K of actual cache and 4K of 4-bit tags to keep track of the cache contents.

PJ: What can you tell us about the cache algorithm?

Dick Barton: It's really very simple—it's not a sequencer. The first time through some code, it is loaded into the cache and executed at near PC speed. On subsequent execution, the code runs out of the cache with no wait states.

PJ: Is the code in the cache contiguous?

Dick Barton: Yes.

PJ: What about compatibility with existing software?

Dick Barton: If it works on the AT, it will work on this card. The AT has forced a lot of standardization in software. The only problems we have seen are with some copy protect schemes—specifically, the scheme that dBASE III first came out with that required that if you execute a loop, you execute it the second time at the same speed. Lotus originally used the same scheme, but it didn't cause any trouble. Ashton-Tate has subsequently modified their use of it.

PJ: Have you been thinking about any enhancements to the board? Retaining the 8088 on the board, for example?

Dick Barton: Let me speak to you from an engineer's standpoint. It may mean security for someone to have the 8088 on the board, from the standpoint of knowing you'll have absolute compatibility with all your software, but we've had very few users who actually have needed it. Also, you can't dynamically switch back and forth between the two processors. You have to shut the machine down. We are working on an enhancement to retain the 8088, though.

PJ: Is it an advantage to be able to retain the 8088 for testing software during development?

Dick Barton: I wouldn't argue with that at all. We plan to bring out a daughterboard soon—in the fall—that will carry the 8088.

Next, we tested a loop of JNZ instructions, with the jump taken at each JNZ:

```
        JNZ SHORT HERE
HERE:
        JNZ SHORT HERE1
HERE1:
        (etc.)
```

286e	286 Express	TinyTurbo	6MHz AT	Compaq Port II	8088
10.0	11.9	11.9	15.37	10.98	27.5

Next, a sequence of PUSH AX
POP AX

Here, the TinyTurbo showed a significant advantage over the Express:

286e	286 Express	TinyTurbo	6MHz AT	Compaq Port II	8088
10.0	33.3	20.2	13.43	9.99	50.6

Next, a benchmark executing a loop that is greater than the size of the short board cache. The loop jumps between two subroutines separated by 14608 bytes. (In addition, each subroutine manipulates the stack, and the loop counter itself is kept on the stack.)

286e	286 Express	TinyTurbo	6MHz AT	Compaq Port II	8088
10.0	14.6	17.6	16.20	11.68	43.8

The Dhrystone Benchmark

The Dhrystone 1.1 benchmark is written in C and measures the performance of a C compiler/processor combination on an average mix of non-floating-point instructions.

Steve Baker compiled the Dhrystone using an Aztec small model optimized for register variables and 286 code.

Dhrystones per second:

Apple IIe	37
IBM PC/XT	275
Macintosh 512	625
IBM PC/AT 6MHz	1225
* Orchid TinyTurbo	1332
* PC Express	1339
MicroVAX II	1385
VAX 11/780	1640
Orchid 286e	1957
Amdahl 5860	28846

Software Development Tasks

Next, some benchmarks of some common software development tasks.

Screen I/O

Write 20,000 characters to the screen with a DOS call:

286e	TinyTurbo	Express	6MHz AT	8088
4.56	21.10	23.24	24*	27.20

*color card, all others are monochrome

File I/O

Load 31991 bytes from a floppy a byte at a time with DOS call:

286e	TinyTurbo	Express	6MHz AT	8088
39.95	63.96	64.73	53.30	152.58

Continued on page 58...

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

PJ: When are you coming out with a 386 accelerator?

Dick Barton: Well, that's a logical question. We've been looking at the chip since November or December — we're a Beta site for Intel — and we have a number of wire wrap boards back in the lab. We're sitting back and watching and waiting. I don't think you'll see us be the first ones out there. Certainly we could demonstrate a 386 accelerator at any time. But it's expensive, and there are still bugs. A 386 accelerator board for the AT would have to cost over $1000.

PJ: What do you think of the 386 so far?

Dick Barton: Well, in the real mode, it is a very good 8088, just as the 286 is a very good 8088. As far as the protected mode goes, we can see those things are there. It will just be a matter of time. There's not a lot of software out there for it yet.

PJ: What sort of performance do you expect from it?

Dick Barton: In the range of 2 to 2.5 as fast as an AT.

Debug Even Faster.

With the Sentinel, hardware debugging technology created especially for Microsoft's new CodeView debugger. Special circuitry unloads CodeView's watchpoint and tracepoint commands onto super-fast hardware. Your debugging time is reduced to seconds, not hours. Find trashed variables and null pointer assignments *fast.*

The Sentinel includes a breakout box to reset your machine and a break button to instantly enter the debugger and see what's going on. Software includes device drivers and our own debugger, Minibug.

The Minibug debugger is available separately. Mini in name, it's maxi in performance, delivering features you'd expect from debuggers costing five times as much. Like symbolic capability, source display, 8087 support and advanced breakpointing features.

Get features that only used to be available in an In-Circuit-Emulator, at a price that won't destroy your budget. If you're like most programmers, the Sentinel will pay for itself in just weeks.

You've got CodeView, the best debugger available. Now make it even better. Buy the Sentinel *today!*

Board, Switch, Software, Manual . . . $275.00
Minibug software only . . . . . $ 39.95

Maranatha Systems, Inc.
8109 133rd Ave NE
Redmond, WA 98052
(206) 881-1068

Trademarks: CodeView - Microsoft Corp.

Circle Number 68 on the Reader Service Card

new! version 3.0

WIZARD C

The following SIEVE benchmark was run without register variable declarations on an IBM/PC with 640K memory and an 8087.

	Exec Time	Code Size	EXE Size
Wizard C 3.0	**: 6.8**	**130**	**7,766**
Microsoft	:11.5	186	7,018
Lattice	:11.8	164	20,068

The new **Wizard C version 3.0** sets new records for all out speed! It leaves other C compilers in the dust! When your project depends on that last ounce of speed, choose Wizard.

Fast executable code, with multiple levels of optimization.

Six memory Models, supporting up to 1 megabytye of code and data, plus mixed model programming.

Effective error diagnosis, including multiple source file cross-checking of function calls.

A comprehensive runtime library, including fully portable C functions, MSDOS and 8086 functions.

8086, 8087, 80186 and 80286 hardware support.

Full Library Source Code included with package.

ROM based application support, including a ROM development package available to create Intel Hex files.

Fully ANSI compatible language features.

"...The compiler's performance makes it very useful in serious software development."
PC Tech Journal
January, 1986

" Wizard's got the highest marks for support."

"The Wizard Compiler had excellent diagnostics; it would be easier writing portable code with it than with any other compiler we tested."
Dr. Dobb's Journal
August, 1985

"...written by someone who has been in the business a while. This especially shows in the documentation."
Computer Language
February, 1985

(617) 641-2379

Only $450.

WIZARD
SYSTEMS SOFTWARE, INC.
11 Willow Court, Arlington, MA 02174

MasterCard VISA

Circle Number 49 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

There's only one winning card in the PC Acceleration Game.

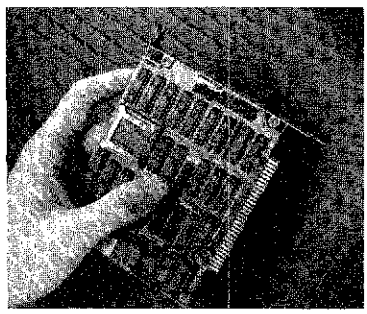
The 286 Express Card™ from PC Technologies

The 286 Express Card™ is unlike any other PC accelerator board. Why? Because it not only gives you all the advantages of a high-performance accelerator, but it also happens to be the original half-slot card. With a price tag too good to believe.

The design of the 286 Express Card is based on Intel's 80286 microprocessor — arguably the fastest VLSI processor in the industry. Designed specifically for the IBM PC and XT, the 286 Express is compatible with existing RAM, communication and peripheral cards, as well as most popular PC DOS software — with no software overlays or modifications required.

The 286 Express Card is loaded with value-added features, including 8KB of cache memory. DMA compatibility. The ability to use the memory on your motherboard and add-on memory cards. An optional 80287 floating point co-processor that works to accelerate the latest versions of spreadsheet and database software. And enough power to speed up your software programs — custom or off-the-shelf — anywhere from 200% to 600%.

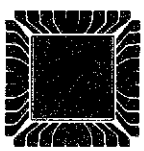
Why gamble with PC productivity? To order — or for the name of the dealer nearest you — call PC Technologies today at 800/821-3086 ext. 103. And let us deal you the winning card.



704 Airport Blvd.
Ann Arbor, Michigan 48104
313/996-9690 • 800/821-3086
Telex 503589

Now compatible with the Tandy 1200*, Leading Edge Model D, and the *new* IBM XT.

*Available through Radio Shack computer centers — Cat. No. 90-2065



IBM, IBM PC and XT are registered trademarks of International Business Machines Corporation.
Leading Edge is a registered trademark of Leading Edge Products, Inc.
Radio Shack and Tandy are registered trademarks of the Tandy Corporation.

Circle Number 75 on the Reader Service Card.

PCT
PC Technologies Inc.

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

PAID ADVERTISEMENT

The 286 Express Offers AT Speed

By Garry Ray

If you're on the verge of tossing your PC in favor of something faster, hold on a minute: There's an add-on coprocessor that will give you much the same performance of an AT. The 286 Express, from PC Technologies, is probably the most useful "speed-up" board that we've seen to date.

The 286 Express gives complete PC compatibility without the mess and bother of start-up device drivers, memory allocation switches or strange performance limitations that often make other boards too oppressive for the general user. In short, this is the board to consider if you want increased speed and utility for your PC.

At the heart of the unit is an 8MHz Intel 80286 microprocessor, supported by a mere handful of ICs and a 40-pin connector (with cable) that plugs into the slot normally occupied by the PC's 8088. In fact, once you've installed the 286 Express, it will act in exactly the same manner as the 8088, albeit faster. With the assistance of on-board Programmable Array Logic (PAL) chips, the processor obtains a clock speed of 7.2MHz from the 14.3MHz color-burst signal on the PC's expansion bus.

Beyond the speed of the 80286, many of the performance improvements of the 286 Express are due to an on-board 8K-byte memory cache that accumulates instructions from whatever program is running. Since the cache is located on the Express board itself, close to the 80286 microprocessor, PC Technologies was able to force continual program execution with no wait states. IBM's AT, which must seek instructions from relatively distant system memory through the 16-bit system bus, runs with one wait state.

The Express continually monitors program flow and swaps segments of code into and out of the memory cache. For example, if the program has a large 'loop' that can execute within the 8K-byte cache, performance for that section of code will be fairly high—in fact, at the physical limit of processing speed for the 80286 running at 7.2MHz.

If, on the other hand, the program has large, non-contiguous code sections that exceed the 8K-byte cache, then performance will be constrained by the requirement to get that code from the PC memory bus into the cache. In any case, performance will always be markedly better than the PC running with an 8088.

Some programs may run into problems with the cache. Since the cache appears to be system memory, there could be conflicts between what memory the program tries to use (system memory) and what memory the program actually uses (cache memory). Other programs, primarily copy-protection schemes that depend upon the contents of pre-defined memory addresses, will become confused by the cache. To surmount these two problems, the user can disable the cache by removing a single on-board jumper pin, which can be replaced after the task has been completed.

Brief Instructions

After opening the 286 Express packaging, you might be inclined to ask, "Where's the rest?" That's because the entire outfit consists of a half-slot PC board, a chip puller and an incredibly brief 12-page installation and instruction manual.

Typically, this sort of product might include a floppy disk with one or more device drivers, a full-slot board loaded with processors, chips and other electronic circuitry and a tome chock-full of rather technical installation and operational instructions. Not so with the 286 Express—it's a simple plug-and-go performer that dispenses with the normal raft of technical considerations.

To get the board running, you simply remove the 8088 with the primitive but functional chip-puller and insert the 286 Express connector. Caution is the key word here, because the chip-puller is more a scaled-down crowbar than anything else. It is not at all difficult to bend the pins of the 8088 using the device. The same is true when installing the circuit replacement; it's relatively easy to bend the pins of the plug.

However, once the board has been installed, it functions as the PC's main—strike that, only—processor at speeds of two to four times greater than the unadorned PC. In fact, with most applications, the 286 Express turns in processing times even greater than IBM's standard 6MHz AT.

In our tests, we measured the performance of the Express against a number of other coprocessor and "speed-up" boards, in addition to the PC, AT and an AT-compatible running at both 6 and 10MHz. (The standard IBM AT runs at 6MHz.)

The results clearly showed that the Express was not only faster than all the PC add-on boards tested by PC Week, but faster than the IBM AT. In fact, most tests showed the Express running at speeds somewhere between those of the AT at 6 and 10MHz. The conclusion of our tests is that the 286 Express turns in speeds roughly equivalent to those of an AT or compatible running at 8MHz.

On the other hand, disk operations are not greatly affected by the Express. This is not a limitation of the board, but rather a physical limitation imposed by the storage devices themselves. If your hard disk runs at 90 milliseconds average access, then that's all you're going to get—no speed-up board can improve disk performance.

However, since the Express will process disk-bound commands (those that control disk access) faster than normal, there will be some measure of improvement in overall performance of disk-intensive programs. Our Disktest, for example, improved by about 15 to 20 percent solely because of the presence of the Express board.

Compatibility with a variety of peripheral boards was well proven in the four weeks we used the Express. In not a single instance did we experience any problems with add-on boards of any type, including some of the new expanded memory boards, modems and local area network adapters. The Express proved ruggedly reliable, causing absolutely no problems with all the software and hardware we tested.

The 286 Express is an elegant solution to a pressing problem—how to get more bang for the buck from a PC. For many users, it could be the alternative to purchasing an AT, bridging the gap between the PC and the next generation of 80386 machines due in 1987.

But whether you intend to upgrade in the future or not, the 286 Express should prove a handsome solution to any concerns about PC performance. ■

PC WEEK DATABOX

Product: 286 Express

Category: PC accelerator board

Who should buy: Those desiring increased performance from an IBM PC or XT

Description: 286 Express adds increased performance to an IBM PC or XT

Distribution: Retail and direct

Company:
PC Technologies Inc.
704 Airport Blvd.
P.O. Box 2090
Ann Arbor, Mich. 48106
(313) 996-9690
(800) 821-3086

286 Express Benchmark

	Sieve	Video Scroll	Squares	Lotus 1-2-3 Multiplication	Payments	Squares
286 Express	13	9	23	4	13	26
PC 4.77MHz	54	11	1:12	10	34	1:25
AT 6MHz	16	7	28	3.5	12	28
AT10 MHz	10	3	17	2	7	17
Orchid PC Turbo	22	5	39	5	16	40

PC Technologies' 286 Express: Turns in speeds comparable to an AT running at 8MHz.

AS SEEN IN PC WEEK
PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Load 31991 bytes from hard disk a byte at a time with DOS call:

286e	TinyTurbo	Express	6MHz AT	8088
31.26	51.76	53.13	49.18	142.14

Load 31991 bytes from a floppy with blockread (1280 bytes at a time) with DOS call:

286e	TinyTurbo	Express	6MHz AT	8088
18.13	18.19	18.24	9.51	18.19

Load 31991 bytes from hard disk with blockread (1280 bytes at a time) with DOS call:

286e	TinyTurbo	Express	6MHz AT	8088
3.68	3.57	3.57	1.15	3.68

(The file I/O figures show the substantial penalty paid when DOS does buffering.)

COMPILE SPEED

Compile in memory

Compile 52525 bytes of source code into 53616 bytes of object code, source and object both in memory:

(2316 lines of code)

286e	TinyTurbo	286 Express	6MHz AT	8MHz AT	8088
10.5	14.5	16.5	17	12.5	44.5

Compile to disk

Compile 52525 bytes of source code into 53616 bytes of .COM file, source in memory, object to disk:

(2316 lines of code)

To hard disk:

286e	TinyTurbo	286 Express	6MHz AT	Compaq	8088
12	14.5	18.5	20	14	44

To floppy disk:

286e	TinyTurbo	286 Express	6MHz AT	Compaq	8088
17	21	24	26.5	22	50.5

CONCLUSION

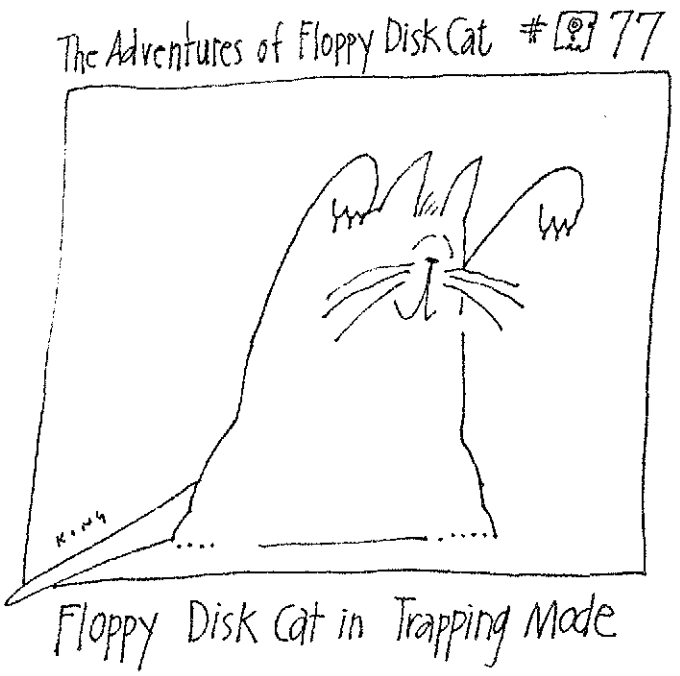
The short boards both performed without a hitch. The AT holds a speed advantage in disk I/O, but the boards were faster than a 6MHz AT on the Dhrystone and when compiling. Both boards far outstripped the standard PC. The TinyTurbo showed a speed edge on real-world tasks, and it retains the 8088 on its board. The PC Express held a slight edge in the Dhrystone benchmark, and costs $100 less at this writing. □

A Turbo Pascal Benchmark Timer

We used the following code to clock a number of the short board benchmarks:

```
var i: integer;
    high1,low1,high2,low2: integer;
    total,total1,total2: real;

begin

inline(
$B4/$00/                        { MOV     AH,00                         }
$CD/$1A/                        { INT     1A                            }
$89/$0E/high1/                  { MOV     [0000],CX                     }
$89/$16/low1                    { MOV     [0000],DX                     }
   );

{put code, procedure or function call to be timed here}

inline(
$B4/$00/                        { MOV     AH,00                         }
$CD/$1A/                        { INT     1A                            }
$89/$0E/high2/                  { MOV     [0000],CX                     }
$89/$16/low2                    { MOV     [0000],DX                     }
   );


if low1<0 then low1:=low1+65536.0;
if high1<0 then high1:=high1+65536.0;
total1:=high1*65536.0 + low1;

if low2<0 then low2:=low2+65536.0;
if high2<0 then high2:=high2+65536.0;
total2:=high2*65536.0 + low2;

total:=(total2-total1)/18.2;

clrscr;
writeln('(test name here)');
writeln('elapsed time=',total:0:2);
```


PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED



FULL POWER DEBUGGING

DSD86, The PC-DOS Debugger 69.95
DSD87, The PC-DOS Debugger with 8087 Support . 99.95
DSD80, The CP/M Debugger 125.00

SoftAdvances

P.O. Box 49473 • Austin, Texas 78765 • (512) 478-4763

1-800-232-8088

Circle Number 33 on the Reader Service Card

ED/ASM-86

Now with Full Screen Editor Built in!

The first truly integrated Editor, Assembler, Debugger, and Linker is now available from Oliver Computing Company.

*** The Editor supports block move, block copy, partial save and partial load insertion. All text is tokenized for both efficiency of storage and very fast assembly time. FAST screen I/O and separate editor/debug windows.**

*** The Assembler can be immediately invoked after any editting command. Output can be directed into memory, to a .COM file, .EXE, or to ED/ASM-86's own efficient .LNK file format. 8087/186/286 instructions supported. Macros and most standard pseudo-ops are supported.**
UNIQUE to ED/ASM is its structured assembly language capability. No more wasted labels and time! Multiple segmented programs can be be sent to .COM files.

*** The built in debugger supports symbolic disassembly, integration with the assembler, as well as disassembly of 8087/186/286 instructions. Display 8087 registers in scientific decimal format. SPLIT SCREEN debugging with a dynamic register window is easily invoked. Disassemble programs to disk file for easier analysis. "Single-step" data editting supported for bytes, words, double words, and 8087 floating constants in memory. You can "patch" assemble a line at a time with symbols from previous assembly (even define symbols as you go!)**

*** The linker supports ED/ASM-86 link files, which can be created from virtually any valid ED/ASM-86 source file, even lines with multiple external symbols in expressions. (Note: not compatible with usual link format)**

*** And many more features too extensive to be listed here.**

All of these features exist in one integrated program. A typical development cycle with ED/ASM-86 consists of entering a program with the editor, or editting an existing program, typing "A:,-IM" to assemble into memory, going into the debugger to immediately test the program. Then you go back to the editor for the next cycle.
ED/ASM was developed for Oliver Computing's own use, and has been used for over a year to enhance itself!

ED/ASM-86 is only $95. Send check or M.O. to:
Oliver Computing Company, P.O. Box 90140, Indianapolis, IN 46290 { (317) 849-4450 }
for immediate delivery via UPS.

Circle Number 15 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

PORTABLE POWER WITH MasterFORTH

Whether you program on the **Macintosh**, the **IBM PC**, an **Apple II** series, a **CP/M** system, or the **Commodore 64**, your program will run unchanged on all the rest.

If you write for yourself, MasterFORTH will protect your investment. If you write for others, it will expand your marketplace.

Forth is interactive – you have immediate feedback as you program, every step of the way. Forth is fast, too, and you can use its built-in assembler to make it even faster. MasterFORTH's relocatable utilities and headerless code let you pack a lot more program into your memory. The resident debugger lets you decompile, breakpoint and trace your way through most programming problems. A string package, file interface and full screen editor are all standard features. And the optional target compiler lets you optimize your application for virtually any programming environment.

The package exactly matches *Mastering Forth* (Brady, 1984) and meets all provisions of the Forth-83 Standard.

MasterFORTH standard package	$125
(Commodore 64 with graphics)	$100
Extensions	
Floating Point	$60
Graphics (selected systems)	$60
Module relocator (with utility sources)	$60
TAGS (Target Applic. Generation System) – MasterFORTH, target compiler and relocator	$495
Publications & Application Models	
Printed source listings (each)	$35
Forth-83 International Standard	$15
Model Library, Volumes 1-3 (each)	$40

(213) 821-4340

MICROMOTION

8726 S. Sepulveda Bl., #A171
Los Angeles, CA 90045

Circle Number 13 on the Reader Service Card

PolyMake *The Leading Make Utility*

Are you still using a prehistoric Make? Now, step up to PolyMake, the most powerful and flexible Make utility available for programmers using MS-DOS. PolyMake is like an intelligent assistant that remembers how to rebuild a program when you change one part of the program. PolyMake will automatically invoke your compiler, assembler, linker, librarian or other tools to update a single program or entire software systems when you type — MAKE. PolyMake comes with built-in rules for rebuilding programs but you can also teach it new rules so you don't have to remember the file dependencies in your program. Advanced programmers prefer capabilities like fully recursive makefile processing and the ability to invoke PolyMake with special "flags" and macros that automate a whole range tasks. New Make users appreciate the Step-By-Step tutorial and intuitive commands. Handles source files written in any language. Requires DOS 2.0 & higher. Compatible with LANs, the IBM PC, XT, AT and other MS-DOS PCs. **For complete details write for the POLYTRON Programmer's Catalog.** $99

PVCS *The Most Powerful & Flexible Source Code Revision & Version Control System.*

The POLYTRON Version Control System (PVCS) allows programmers, project managers, librarians and system administrators to effectively control the proliferation of revisions and versions of source code in software systems and products. PVCS is a superb tool for programmers and programming teams. (A special LAN version is also available.) If you allow simultaneous changes to a module PVCS can merge the changes into a single new revision. If changes conflict, the user is notified. Powerful capabilities include: Stores and retrieves multiple revisions of text; Maintains a complete history of revisions to act as an "audit trail" to monitor the evolution of a software system; Maintains separate lines of development or "branching"; Provides for levels of security to assure system integrity; Uses an intelligent "difference detection" to minimize the amount of disk space required to store a new version. Requires DOS 2.0 or higher. Compatible with the IBM PC, XT, AT and other MS-DOS PCs. Single User version $395. 5-station LAN version $1,000, add $500 for each additional 5 stations. $395

TO ORDER: VISA/MC 1-800-547-4000 Dept. No. 306; Oregon/Outside US, 503-684-3000
Send Checks/POs To: POLYTRON Corp. 1815 NW 169th Pl., #2110, Dept. No. 304, Beaverton, OR 97006

POLYTRON
High Quality Software Since 1982 ®

Circle Number 26 on the Reader Service Card

The Adventures of Floppy Disk Cat #23

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

The E-Heart Graphics Card $250
E-Heart Engineering
Irvine, CA
(714) 261-1725

The E-Heart is a graphics coprocessor board that works in conjunction with a Color Graphics Adapter. The CGA output goes to the E-Heart instead of the monitor, and the E-Heart connects to the monitor.

The E-Heart uses an HD63484 chip, an intelligent graphics generator that is controlled from a program through four bytes of I/O ports. It adds graphics to the CGA picture passing through. The board has a driver that interfaces with Turbo Pascal, and it is fast enough for real-time animation. It will draw a million color pixels per second and up to 50,000 vectors per second. It will simultaneously display 16 colors at a resolution of 640 x 200, user upgradable to 640 x 400, has a hardware generated cross-hair cursor, independent x and y zoom capability with factors from 1 to 16, software selectable hardware clipping, and hardware support for line, polygon, arc, paint, dotted lines, fast fill, rectangle, circle, elliptic arc, pattern, patterned paint, dithered paint, polyline, ellipse, filled rectangle, copy, and draw dot.

E-Heart Engineering is looking for individual purchasers and OEMs.

Advanced MS DOS
by Ray Duncan
Microsoft Press
Redmond, WA
(800) 426-9400

A brand-new 468-page tome that covers disk file and record operations, disk directories and volume labels, DOS disk internals, memory management, EXEC functions, installable device drivers, DOS functions and interrupts, error codes, notes on version-to-version differences, assembly language examples, BIOS video services, and the EMS specification. It's subtitled "The Microsoft Guide for Assembly Language and C Programmers." It's thorough.

SMALLTALK/V $99
Digitalk Inc.
Los Angeles, CA
(213) 645-1082

Smalltalk with bit-mapped graphics, windows, mouse support, a built-in Prolog compiler, object-swapping virtual memory, and a source-level debugger. Not copy protected, requires 512K, a Microsoft or compatible mouse, and a CGA, EGA, or AT&T graphics controller. Concurrent with this release, Digitalk's original non-graphic Smalltalk implementation, Methods, has been reduced to $79 and has been bundled with a communications interface to UNIX systems that is optional with Smalltalk/V.

According to the president of Digitalk, the Prolog compiler was included to enable programmers to move freely from object-oriented to rule-based programming.

ELIZA
The Software Toolworks
14478 Glorietta Drive
Sherman Oaks, CA 91423
(818) 986-4885

For those (including myself) who never had a chance to look at Joseph Weizenbaum's 1966 classic, *ELIZA*, here it is, enhanced by Walt Bilofsky, the chairman of Software Toolworks.

The program takes 40K and includes Weizenbaum's original *script*, an extension of his original script interpreter, and a new script. The program algorithm is thoroughly described, and one can easily tinker with new scripts.

The package also includes Weizenbaum's original 1966 research paper on the subject.

I understand that Weizenbaum wrote the program (which attempts to be a non-directive psychiatrist) to show how ridiculous the idea of A.I. was. He was chagrined to find people to whom he had shown the program returning to say, "Um, could I talk to that computer program again?"

Here's a conversation, with computer response in caps:

Men are all alike.
IN WHAT WAY
They're always bugging us about something or other.
CAN YOU THINK OF A SPECIFIC EXAMPLE
Well, my boyfriend made me come here.
YOUR BOYFRIEND MADE YOU COME HERE
He says I'm depressed much of the time.
I AM SORRY TO HEAR YOU ARE DEPRESSED
It's true. I am unhappy.
DO YOU THINK COMING HERE WILL HELP YOU NOT TO BE UNHAPPY
I need some help, that much seems certain.
WHAT WOULD IT MEAN TO YOU IF YOU GOT SOME HELP
Perhaps I could learn to get along with my mother.
TELL ME MORE ABOUT YOUR FAMILY
My mother takes care of me.
WHO ELSE IN YOUR FAMILY TAKES CARE OF YOU
My father.
YOUR FATHER
You are like my father in some ways.
WHAT RESEMBLANCE DO YOU SEE
You are not very aggressive but I think you don't want me to notice that.
WHAT MAKES YOU THINK I AM NOT VERY AGGRESSIVE
You don't argue with me.
WHY DO YOU THINK I DON'T ARGUE WITH YOU
You are afraid of me.
DOES IT PLEASE YOU TO BELIEVE I AM AFRAID OF YOU
My father is afraid of everybody.
WHAT ELSE COMES TO MIND WHEN YOU THINK OF YOUR FATHER
Bullies.
DOES THAT HAVE ANYTHING TO DO WITH THE FACT THAT YOUR BOYFRIEND MADE YOU COME HERE

Modula-2
Turbo Translator $49.95
Make Utility $29
Windowing Package $49
Logitech Inc.
Redwood City, CA
(415) 365-9852

Logitech has announced three additions to its Modula-2 development system—a Turbo Pascal to Modula-2 translator, a Make utility, and a Windowing package.

The Turbo Translator comes bundled with a number of new Modula-2 library modules, including screen handling, graphics, and file handling, that have been adapted to correspond directly to Turbo Pascal functions.

The Make Utility runs through a program after changes have been made to the source file and selects and handles modules affected by the changes.

The Windowing Package features virtual screens, color support, and overlapping windows and uses only 15K of code.

Continued . . .

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

COMBINE THE
RAW POWER OF FORTH
WITH THE CONVENIENCE
OF CONVENTIONAL LANGUAGES

HS/FORTH

Why HS/FORTH? Not for speed alone, although it is twice as fast as other full memory Forths, with near assembly language performance when optimized. Not even because it gives MANY more functions per byte than any other Forth. Not because you can run all DOS commands plus COM and EXE programs from within HS/FORTH. Not because you can single step, trace, decompile & dissassemble. Not for the complete syntax checking 8086/8087/80186 assembler & optimizer. Nor for the fast 9 digit software floating point or lightning 18 digit 8087 math pack. Not for the half megabyte LINEAR address space for quick access arrays. Not for complete music, sound effects & graphics support. Nor the efficient string functions. Not for unrivaled disk flexibility — including traditional Forth screens (sectored or in files) or free format files, all with full screen editors. Not even because I/O is as easy, but far more powerful, than even Basic. Just redirect the character input and/ or output stream anywhere — display, keyboard, printer or com port, file, or even a memory buffer. You could even transfer control of your entire computer to a terminal thousands of miles away with a simple >COM <COM pair. Even though a few of these reasons might be sufficient, the real reason is that we don't avoid the objections to Forth — WE ELIMINATE THEM!

Public domain products may be cheap; but your time isn't. Don't shortchange yourself. Use the best. Use it now!

HS/FORTH, complete system: $395. with "FORTH: A Text & Reference" by Kelly and Spies, Prentice-Hall and "The HS/FORTH Supplement" by Kelly and Callahan

Visa Mastercard

HARVARD SOFTWORKS
PO BOX 69
SPRINGBORO, OH 45066
(513) 748-0390

Circle Number 42 on the Reader Service Card.

Mach 2 $75
MicroHelp, Inc.
Marietta, GA
(404) 973-9272

MicroHelp, publisher of "The Inside Track" and "Peeks and Pokes," has released "Mach 2," an additional toolbox of Assembler and BASIC subroutines for BASIC programmers. It's compatible with Quick BASIC, BASIC/BASICA, and GW-BASIC.

Calls to subroutines use the same syntax whether your program is compiled or interpreted. Assembler subroutines are loaded outside BASIC's data area. Mach 2 contains Lotus style menus, read/write, string, display, print using, scroll, rounding, and DOS and BIOS call routines.

C- Display Compiler $125
C- Display Librarian $145
both $245
SYDETECH
System Development Technologies
Flushing, NY
(718) 886-0221

SYDETECH has released Version 3.0 of C-Display Utilities, C software tools for interactive and graphics programming.

The C- Display Compiler is a screen management utility with full screen and window display forms. It supports colors and character-generated graphics.

The C- Display Librarian executes BIOS video services, virtual display management, bit-mapped graphics, and interactive screen I/O. Included is part of the source code and programming examples.

The software is available for the PC/XT/AT, or TI Pro, Lattice 2.15/3.0, and Microsoft 3.0.

Owners of the earlier SYDETECH versions, which were marketed for the TI, are entitled to a free upgrade.

ACTOR
The Whitewater Group
Evanston, IL
(312) 491-2370

Charles Duff, who wrote the Neon language for the Mac, is writing ACTOR for the PC, an object-oriented, interpreted language that is integrated with the Microsoft Windows environment. It's scheduled for an October release.

System V/AT
Runtime System $160
Software Development System $169
Text Preparation $169
All Above $439
Three to Eight User Upgrade $99
Microport
Scotts Valley, CA
(408) 438-UNIX or 800/PC2-UNIX (outside CA)

A multi-user/tasking AT version of UNIX System V/286 Release 2, System V/AT comes with more than 200 utilities, the symbolic debugger *sdb*, the shell-layering job-control facility and the F77 Fortran compiler. It supports the 286's 16 Megabyte virtual address space. The Microport staff was involved in the standard System V/286 port for Intel and used that experience to develop System V/AT.

International Resource Development Inc.
Market Research—Product Planning
Norwalk, CT
203) 866-7800

A News release from IRD:

Circuit Boards Soon To Wrap Around Human Bodies?
Advances in circuit board fabrication will lead to the possibility of designing custom made "intelligence enhancers" that will enclose themselves around parts of the human body. In fact, eventually, as these computer systems become more and more "intelligent," the human body itself may be viewed as parasitic by the new life form molded in the image of ourselves.

Think we can just slink away, and they won't notice? ⌴

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

PJ BILLBOARD

PJ Billboard is a product and services directory containing listings for hardware and software of interest to the IBM PC programmer. The goal is to provide our readers with a wide range of information on current products and updates. PJ Billboard serves our advertisers by providing a low cost, flexible format in which they can present their products to our readers.

The standard layout is four columns, 9 picas wide, with copy set in 7 point on 8 point leading. PJ Billboard offers a 400-character product description, including your company name, address, and telephone number. Boldface and italics are available.

The listings are $65, prepaid, non-commissionable for a one-time insertion. Listings must be received typed, preferably on one of our convenient order forms. Copy deadlines are the same as the normal ad materials closing dates, which can be found on our rate card, or you may call the Programmer's Journal Advertising Department at 503-484-0602 for more information.

PageMaker Services
User manuals, Advertising, Packaging and Direct Mail
- Graphic production
- Design and illustration
- Editing and copy writing
- Download your existing documents
- Professional quality work
- Fast turnaround

Ralph Bentley
Blackbird Studios
888 Armstrong Ave.
Eugene, OR 97404
(503) 689-6651

CSL, C Scientific Library is available for either the Computer Innovations or Microsoft C compiler. Over 300 functions and 556-page manual. Superior to any other such library. Includes implementations of many advanced algorithms. (send $2.50 for complete annotated index of CSL functions). $245.00

Ligenware Technologies
13090 La Vista Drive
Saratoga, CA 95070
(408) 867-1184)

The SCREEN GENERATOR
SAVE TIME! Powerful Screen Designer and Memory Resident Screen Manager eliminates tedious screen management tasks from your program. Handles all screen display & data entry. Easy access to Screen Manager from your program. Not a Code Generator! NO Royalties, Not Copy Protected. Easy conversion from other sytems. Use with BASIC, TURBO ($59), FORTRAN, C, Realia COBOL, BetterBASIC, PL/M-86, or Assembler. Call if not listed. Price $125.

The West Chester Group
P.O. Box 1304
West Chester, PA 19380
(215) 644-4206

Develop Microprocessor Software
AVMAC® macro cross assemblers include **AVLINK**®, **AVLIB**®, **AVREF**®, for MS/PCDOS, $349, VAX/UNIX $995. It's easy to turn your PC into a complete low-cost development system by adding our **AVSIM**® full-screen simulator/debugger, $299. **TRICE**® self-powered in circuit emulators from $498. **AVPROM**® EPROM programmers from $398.

AVOCET SYSTEMS INC.
P.O. Box 490-PJ
Rockport, ME 04856
(800) 448 8500

Advanced Programming Tools
The C Library — Over 140 macros, functions, and programs. $29.95
The C to DOS Interface — Over 70 routines to control DOS and the ROM-BIOS. $59.95
The FORTRAN to DOS Interface — Over 70 routines to control DOS and the ROM -BIOS. $59.95
The Turbo Pascal Programmer's Library — Over 175 essential routines for Turbo Pascal. $59.95

Please include $4.00 shipping. For more information write to:

Kris Jamsa Software, Inc.
Box 26031
Las Vegas, NV 89126

SEARCH by CompuMagic
Find what you're looking for *FAST!* SEARCH program or text files for one or more strings. See context for each match. On-line help; easily-learned wildcards and boolean operators; I/O redirection. Compatible with most word processors. Use MSDOS paths, too. Not copy-protected. With documentation and 30-day guarantee, just **$30.**

CompuMagic, Inc.
P.O. Box 437
Severn, MD 21144
(301) 969-8068

MS-DOS 2.11 & GWBASIC
500 Copies at $25 ea. takes all or $40 ea./singles. One of California's largest liquidators has inventory in excess of 500K pieces of software priced 50% to 90% off. There has to be something for you among all these programs. At these prices can you afford not to call? (We also purchase quantities of ANYTHING at the right price.)

AT&T Liquidators
2640 S. Harbor Blvd.
Santa Ana, CA 92704
(714) 751-2667

X-TOOLS
Eliminate paper and pencil flow charts! Use X-TOOLS to create and edit structograms on-screen or convert existing code to structograms, then generate source text and precompile your programs. X-TOOLS run on IBM-PC and compatibles; DEC Rainbow, Microvax and Vax; and Intel; under PC-DOS, MS-DOS, VMS, CP/M, CCP/M, etc. Languages covered include: C, BASIC, Pascal, PL/M, Fortran 77 and COBOL.

Software Design Tools, Inc.
6290 Sunset Blvd., Suite 1126
Los Angeles, CA 90028-8770
(213) 463-6102

FORTRAN PROGRAMMERS—
Complete graphics libraries callable from MS, R-M/IBM Profort or Lahey F77L. GRAFMATIC (Screen graphics) includes tabular, function plots, contours, bars, pies, log/linear axes and more. 3D solid models and wire frame. Supports EGA,CGA or Hercules Mono Boards. PLOTMATIC drives H-P or Houst. Inst. plotters. Write or call for info.

MICROCOMPATIBLES
301 Prelude Drive, Dept. PJ
Silver Springs, MD 20901
(301) 593-0683

Seeking new challenges? New experiences? C Associates, the UNIX/C specialists, are currently seeking professionals skilled in any of these areas: UNIFY/Informix database projects, Software Engineering, Kernal Modification, Data and Voice Communications, and Pre & Post Sales Support Positions.

We offer individual, no hassle counseling, and we are concerned with your personal goals. Positions available in Washington, DC and nationwide.

Please call or send resume to:
John Capozzi, Jr., CPC
C Associates, Suite C
1364 East Capitol Street, N.E.
Washington, DC 20003-1533
(202) 544 0821

DEVELOPERS, and MARKETERS: SAYWHAT?! is for you! Develop data entry and display screens, write them to video ram, put together a slide show. Menu access to *all* IBM screen characteristics. **Saywhat?!** is *not* copy protected. *No* additional license fees. Send $44.95 to

PSDW
1328 Magnolia Ave.
San Carlos, CA 94070
(415) 593-6431
CA residents add 6.5% sales tax.

CROSS ASSEMBLERS FOR MS-DOS SYSTEMS! Turn your IBM PC or compatible into a microprocessor development station—develop code for: 6301, 6502, 6800, 6801, 6805, 6809, 8080 and Z80 (Inquire about others).
FAST—INEXPENSIVE! only $49.95 (Calif. residents add 6.5% tax)

STS ENTERPRISES
5469 Arlene Way
Livermore, CA 94550
(415) 455-5358
(Sorry no Credit Card orders)

Documentation
- Writing, editing, formatting for photocopying or typesetting
- MS DOS, Apple, AT&T 3B series UNIX
- Over thirty manuals
- Credits include
  Dave Winfield's BATTER UP!
  (Billboard Top Twenty)
  Action Inn Management
  (AT&T 3B2)
  SmartCAM
  (MS DOS, Apple)
  Programmer's Journal
  (Managing Editor)

Erin O'Connor
c/o Programmer's Journal
P.O. Box 30160
Eugene, OR 97403
(503) 484-2162

POWER TOOLS(tm)
IBM PC Resident DOS Utility
Many more features than "Norton"... memory residency like "Sidekick"... plus a Disk Optimizer to speed-up your Hard Disk... all for ⅓ the cost. Execute DOS commands from inside your application program. Recover lost data. Copy/Move files with Date/Time protection. Runs on PC, XT, AT with 256K+. Not Copy protected. $53. Ppd. USA. Call/Write.

MLI MICROSYSTEMS
PO. Box 825, Dept. PJ
Framingham, MA 01701 USA
(617) 926-2055

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

Housekeeping

ERRATA

Hal W. Hardenbergh called to complain that we gave him too long a drive to work last issue (July/August 1986). Digital Acoustics Inc., is in Santa Ana, not Santa Clara, CA.

Bill Hunt asked us to let readers know that the nineteenth line in the code for "Memory Resident Programs in C" (July/August 1986, p. 46) should read

```
return( dreg.x.ax );  /* return DOS error code */
```

Bernard H. Robinson, Jr. wrote to correct our typographical errors in "Tips on Formatting with BASIC PRINT USING Statements" in the July/August issue (Volume 4, Number 4) of *PJ*. The format of the BASIC PRINT USING statement near the top of the second column on page 50 should be

```
PRINT USING F$; list [;]
```

In "Figure 1. Output from P-USG.BAS" on page 54, the words "TO DATE" should appear at the end of the first line of the summary, not at the beginning of the second line as shown.

Rank the articles in this issue of PJ. Refer to the list below for the article letters, and grade the articles on the postage paid Reader Service Card, Number 1 the best, and so on.

A ____PJ FORUM: DTACK GROUNDED, Hal W. Hardenbergh
B ____REDIRECTIONS: Frank D. Greco
C ____PROLOG: An Introduction, Mark Minasi
D ____Turbo Prolog: It's Not Real Prolog, Lew Baxter
E ____Turbo Prolog: Another Miracle? Tom Swan
F ____CONSULTING: Tax Traps, Consulting Styles, Nest Eggs, Paul Barkley
G ____ASM: Search and Display, Dan Daetwyler
H ____ASM: An Assembler Programming Suggestion, Graham Pearson
I ____C: The Proposed ANSI C Standard, Part 3, Rex Jaeschke
J ____C: Monitoring Stack Usage and a Guide to Startup, William Hunt
K ____REVIEW: Short Cards

ADVERTISER INDEX

CIRCLE NUMBER		PAGE NUMBER
53	Allen, Emerson & Franklin	22
1	ATRON	8
30	Austin Code Works	49
55	Barrington Systems	32-33
2	Blaise Computing	7
3	Borland International	29
38	C-Ware	47
79	Chalcedony Software	21
56	Compuview Products, Inc.	cov 2
78	Conversational Computer Syst.	22
40	Creative Prog. Consultants	4
66	Custom Software Systems	13
5	Dair Computer Systems	34
6	Data Base Decisions	42
41	Design Solutions	23
67	E-Heart Engineering	15
8	Ecosoft, Inc.	cov 4
76	Emerald Computers International	51
9	Entelekon	36
11	Gimple Software	36
42	Harvard Softworks	62
57	Hawkeye Grafix	38
58	Interface Group	17
59	L.U.C.I.D.	16
63	Laboratory Microsystems, Inc.	14
44	Lahey Computer Systems	30
68	Maranatha Software	55
69	MDS, Inc.	4
13	Micromotion	60
70	Micromotion	15
65	Mnematics Videotex	4
15	Oliver Computing	59
16	Opt-Tech Data Processing	31
75	PC Technologies	56-57
71	Pecan Software Systems, Inc.	9
17	Peter Norton Computing	35
26	Polytron	60
31	Quantum Information Systems	30
46	Quilt Computing	34
20	Rational Systems	35
33	Soft Advances	59
72	Speedware	39
74	Sydetech	50
35	Turbo Source Search	23
25	Turbopower Software	39
73	Whitewater Group	13
49	Wizard Systems Software	55

To receive advertising information, circle #89

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

T-SHIRTS

BORN TO PROGRAM PJT101

Show your true colors when cruisin' the local computer store. Let 'em know that programmers rule in an authentic black T-shirt with this hot multi-color design.

PJ LOGO PJT100

For those of you who care enough to wear the very best, we offer our world famous PJ logo printed on the T-Shirt color of your choice: White, Lt. Blue, Beige or Gray.

PARALLEL PORTS PJT102

These twin vessels are guaranteed to enhance communications with your favorite printer or anyone else you dump your text to. Multi-color imprint on a wine-colored T-Shirt.

NOW! SOFT WEAR FROM PROGRAMMER'S JOURNAL!
Only $9.95 each, postage paid.

By popular demand, your favorite computer jokes and cover graphics by Julia King and Ralph Bentley come alive in dynamic multi-color imprints on high quality 50/50 cotton T-Shirts. Use the form on this page to order your own unique, custom soft wear. Fill out and mail today with your payment to:

Programmer's Journal
29 W. 29th Ave., Suite 5
Eugene, OR 97405

RS-232 HEAD PJT103

Here's a connection you can't refuse. Plug your head into this neat-as-a-pin computer gray T-shirt with colorful high-tech imprint.

COMMON C BUG PJT104

Everyone's got 'em! Now proudly display your own specimen. The bug is brown (naturally), crawling across a gold nameplate on a beige T-shirt.

SIZES AVAILABLE: S, M, L, XL

DESIGN NO.	SIZE	COLOR	QTY	AMOUNT	SEND TO:
					NAME
					COMPANY
					ADDRESS
					CITY STATE ZIP
					CHECK ENCLOSED ☐ CHARGE MY: VISA☐ MC☐
TOTAL ENCLOSED				$	CARD NO. EXP. DATE

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

DR. JACK PURDUM

Get the proven product from the man who wrote the books on "C".

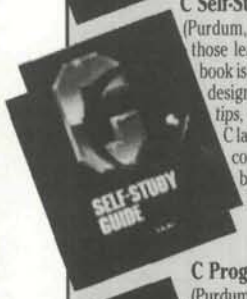
C Programming Guide

After reading the 1st edition, Jerry Pournelle (**BYTE** Magazine) said: "I recommend this book...Read it *before* trying to tackle Kernighan and Ritchie." The second edition expands this best seller and walks you through the C language in an easy-to-understand manner. Many of the error messages include references to this book making it a perfect companion to Eco-C88 for those just starting out with C.

$20

C Self-Study Guide

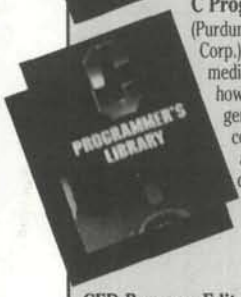
(Purdum, Que Corp.) Designed for those learning C on their own. The book is filled with questions-answers designed to illustrate many of the *tips, traps, and techniques* of the C language. Although written to complement the Guide, it may be used with any introductory text on C.



$17

C Programmer's Library

(Purdum, Leslie, Stegemoller, Que Corp.) This best seller is an intermediate text designed to teach you how to write library functions in a generalized fashion. The book covers many advanced C topics and contains many useful additions to your library including a complete ISAM file handler.



$22

CED Program Editor

CED now supports on-line function help. If you've forgotten how to use a standard library function, just type in the name of the function and CED gives you a brief summary, including function arguments. CED is a full screen editor with auto-flagging of source code errors, multiple windows, macros, and is fully configurable to suit your needs. You can edit, compile, link, and execute DOS commands from within the editor. Perfect for use with Eco-C88. For IBM PC, AT and look-alikes.

$29.95

C Source for Standard Library

Contains all of the source code for the library functions that are distributed with Eco-C88, excluding the transcendentals and functions written in assembler.

$10 ($20 if not with order)

Developer's Library

Contains all the source code for the standard library, including transcendental and assembler functions. Available with compiler purchase only.

$25 ($50 if not with order)

Ecosoft Inc.
6413 N. College Ave.
Indianapolis, IN 46220

THE FIRST PROFESSIONAL 'C' COMPILER FOR UNDER $60.00

Limited Time Offer
FREE
CED TEXT EDITOR WITH "BUILT-IN" FUNCTION HELP.

NOTHING IN THIS PRICE RANGE EVEN COMES CLOSE.

"Ecosoft's Eco-C is going to turn some major heads."

"Eco-C is the first compiler reviewed that has clearly begun implementing the coming ANSI standard. Eco-C supports prototyping."

"Eco-C performed well on all the benchmarks, generating code that was quite comparable to that of compilers 10 times as costly."

from:
Christopher Skelly
Computer Language,
Feb., 1986

"The driver program is another strength: it compiles and links a list of files and provides a simple MAKE capability."

"This compiler does handle syntax errors much better than average—no avalanche of spurious messages here."

from:
William Hunt
PC Tech Journal,
Jan., 1986

"Eco-C88 is a high-quality package... comparable to systems costing much more."

"Eco-C88 is one of the fastest..."

"It is convenient to use, works well, and produces acceptably compact and fast programs."

from:
Dr. David Clark
Byte, Jan., 1986

Minimum System Requirements:

To use Eco-C88 Release 3.0, you must have:

1. An IBM PC, XT, or AT-compatible computer with monitor.
2. 256K or more memory.
3. Two 360K disk drives, or a hard disk.
4. PC or MSDOS 2.1 or later to include the MSDOS linker.

Eco-C88, mini-make, memfiles and CED are trademarks of Ecosoft Inc.
IBM is a trademark of International Business Machines.
UNIX is a trademark of Bell Labs.
MSDOS and MASM are trademarks of Microsoft.

Full-featured C compiler. Supports all C features, data types (except bit fields), and operators.

New Language Enhancements. You also get prototyping, enum and void data types, plus structure passing and assignment.

Tiered Error Checking. All syntax errors are automatically flagged, but you can select the level of "link-like" semantic error checking you want.

Complete Standard Library. Over 200 functions, many of which are System V compatible for greater source code portability.

Screen and Memory Functions. Now you can write programs that use color, cursor addressing, even ones that let you design your own graphics functions. You also get *memfiles™ that allow you to access* memory outside the normal data segment as a file.

8087 and 80287 Support. If you have one of these math chips, your programs will take immediate advantage of it. If you don't have one, the code automatically switches to software floating point.

Full Screen Editor. The CED editor is a full screen editor with multiple windows, macro commands, on-line function help, plus a full set of editing commands. (Requires a true IBM PC compatible.) You can edit, compile, link, and execute programs from the editor which greatly reduces development time.

Includes a cc and mini-make™. The UNIX-like cc makes compiling programs a snap. You can run cc from within the CED editor.

ASM or OBJ Output. You can select assembler or relocatable output from the compiler. Both are MASM compatible and ready for use with the MSDOS linker to produce EXE files.

	Eco-C88	Lattice	Computer Inn. C86	Microsoft	Mark Williams
sieve	12	11	13	11	12
fib	43	58	46	109	—
deref	14	13	—	10	11
matrix	22	29	27	28	29

Computer Language, Feb., 1985, p. 79. Reproduced with permission.

Orders only: 1-800-952-0472

ORDER FORM CLIP & MAIL TO: Ecosoft Inc., 6413 N. College Ave., Indianapolis, IN 46220

- ☐ C Compiler $59.95 _____
- ☐ C Programming Guide $20.00 _____
- ☐ C Self-Study Guide $17.00 _____
- ☐ C Programmer's Library $22.00 _____
- ☐ CED Program Editor $29.95 _____
- ☐ C Source for Standard Library $10.00 ($20.00 if not with order) _____
- ☐ Developer's Library $25.00 ($50.00 if not with order) _____

SHIPPING _____ $4.00

TOTAL (IND. RES. ADD 5% TAX) _____

PAYMENT: ☐ VISA ☐ MC ☐ AE ☐ CHECK

CARD # _____ EXPIR. DATE _____

NAME _____

ADDRESS _____

CITY _____ STATE _____

ZIP _____ PHONE _____

Circle Number 8 on the Reader Service Card

PRODUCED 2004 BY UNZ.ORG
ELECTRONIC REPRODUCTION PROHIBITED

ECOSOFT